M.Sc. Thesis
Approximating Averages of Geometrical and Combinatorial Quantities

by Kfir Barhum
Advisor: Prof. Oded Goldreich
Department of Computer Science and Applied Mathematics
Weizmann Institute of Science

February 2007

Abstract

We look into the problem of estimating the average of quantities relating to combinatorial and geometrical objects. In all the cases we study it is possible to get the exact average of these quantities by using the trivial algorithm that calculates the average by obtaining all the quantities. Using the underlying structure of these problems we are able to obtain faster algorithms that approximate the average. Specifically, we consider randomized algorithms that are given an approximation parameter ε, and return a $(1+\varepsilon)$-approximation of the average quantity. Our work focuses on the problem of calculating the average degree of a hypergraph, and the problem of calculating the average distance between every pair of points in a set of points in the d-dimensional Euclidean space.

To My Parents

Acknowledgments

I am grateful to my advisor, Oded Goldreich, for many indispensable discussions, his endless patience, and his kind support during the last two years at the Weizmann Institute. During this time Oded has had significant influence on my perception of Theoretical Computer Science. I wish to thank him for his advice both in my personal and in my professional life. I have enjoyed much his special sense of humour and his opinions on many varying subjects. I would like to thank my friend Gilad Tsur for our many interesting conversations.

Thank you to my family and friends who supported my throughout the years.

1 Introduction

> He also made the large bronze basin called "The Sea". It measured 15 feet from rim to rim, was circular in shape, and stood seven-and-a-half feet high. Its circumference was 45 feet.
>
> *1 Kings, 7:23*

We deal with the problem of estimating the average of quantities relating to combinatorial and geometrical objects. In the cases we study it is possible to get the exact average of these quantities by using the trivial algorithm that calculates the average by obtaining all the quantities. Although these algorithms obtain exact values, at times, when considering very large inputs, they may be unfeasible and one may look for alternatives. In our work, we exploit the underlying structure of it in order to obtain a faster algorithm that approximates the average. In each case, specific method and analysis is being used.

In general, estimating the average of function describing arbitrary quantities on n points cannot be done in non-trivial time. Consider for example, an algorithm trying to evaluate the average of one of the following functions: The first function is the constant function $f \equiv 1$, whereas the second function would be the value 1 on its entire domain except for one (random) point on which it assumes the value $n \cdot B$ for some large value B. It takes $\Omega(n)$ queries to distinguish between the functions. Note that their averages are 1 and B correspondingly, making it is impossible to estimate well the average without distinguishing these functions. Similar examples exist even when the range of f is bounded (say by n).

Organization and Summary In section 2 we study the problem of calculating the average degree of a hypergraph in the case where each edge connects at most k vertices. Section 3 deals with the problem of calculating the average distance between every pair of points in a set of points in the d-dimensional Euclidean space. During our work on this problem, we came up with an algorithm for approximating the diameter of such a set. We later found that a very similar algorithm was already known. We survey briefly this algorithm in Section 4. Each section is accompanied with a short introduction to the problem it deals with, where we introduce the problem, survey relevant previous work and present our results.

Preliminaries As the basic definition of approximation algorithms, we use the following standard one: For $\alpha > 1$, a α-approximation of a quantity $q : \{0,1\}^* \to (0,\infty)$ is an algorithm that on input x, with probability at least $2/3$, outputs a value in the interval $[q(x), \alpha \cdot q(x)]$.

Our algorithms will all be uniform in the sense that we actually present an algorithm that takes ε as a parameter and output a $(1+\varepsilon)$ approximation of a quantity. For simplicity of presentation we allow the algorithm to output a value in the interval $[(1-\varepsilon) \cdot q(x), (1+\varepsilon) \cdot q(x)]$. Indeed, the output can be "normalized" by division (by $1-\varepsilon$). This is due to the fact that the our algorithms have polynomial dependence on $\frac{1}{\varepsilon}$ and may be given $\frac{\varepsilon}{2}$ as their input.

Note, that the error probability can be decreased to 2^{-p} by invoking the basic algorithm for $O(p)$ times and outputting the median value. In this sense, the error probability of $\frac{1}{3}$ is arbitrary, and every constant smaller than $\frac{1}{2}$ will do.

2 Average Degree of (up to) K-Uniform-Hypergraphs

2.1 Introduction

We look into the problem of approximating the average degree of hypergraphs with constant size hyperedges.

Recall that a hypergraph is an ordered pair $H = (V, E)$, where V is a set of vertices and E is a set of hyperedges. That is, every hyperedge $e \in E$ is a subset of the vertices containing at least 2 vertices (no loops or double hyperedges are allowed).

A k-uniform hypergraph, is a hypergraph where each hyperedge contains k vertices (i.e. $\forall e \in E$ it holds that $|e| = k$). A (simple) graph can be viewed as a 2-uniform hypergraph. A vertex degree $d(v)$ is the number of hyperedges that v is member of. Denoting $|V| = n$, we are interested in approximating $\bar{d} \triangleq \frac{1}{n}\sum_{v\in V} d(v)$. Note that for a k-uniform hypergraph, it holds that $\bar{d} \cdot n = k \cdot |E|$.

We consider algorithms for estimating $\bar{d}$ that are allowed to perform two types of queries: *degree queries* and *neighbor queries*. Respectively, for any vertex v of its choice, the algorithm can obtain $d(v)$, and for any v and $j \leq d(v)$, the algorithm can obtain the j^{th} hyperedge of v.

Previous Work The task of approximating the average degree of a (simple) graph was studied by Feige [1], and Goldreich and Ron [2]. Whereas both works show a $(2+\varepsilon)$-approximation to the average degree of a graph using only degree queries, the latter establishes a $(1+\varepsilon)$-approximation using neighbour queries. Specifically, their algorithm uses $\tilde{O}((n/\bar{d})^{\frac{1}{2}} \cdot poly(\frac{1}{\varepsilon}))$ neighbour queries.

Our Result We show how to extend the algorithm suggested by Goldreich and Ron to the case of hypergraph with edges that connect at most k vertices. We call such hypergraphs up-to-k-uniform hypergraphs. Our algorithm achieves a $(1+\varepsilon)$-approximation of the average-degree d of such hypergraphs using $\tilde{O}((n/\bar{d})^{\frac{k-1}{k}} \cdot poly(\frac{k}{\varepsilon}))$ neighbour queries. Indeed, the case of $k = 2$ coincides with the algorithm presented by Goldreich and Ron.

2.2 An Overview of the Algorithm

Recall that $\bar{d} = \frac{1}{n}\sum_{v\in V} d(v)$. Since every hyperedge contributes k to the sum of all degrees, we may consider each hyperedge as k *directed hyperedge*s, connecting each of its vertices to all the others. Denoting such a directed hyperedge as $(v_i, \{v_1, \ldots, v_k\})$, we are interested in estimating the size of the set of all directed edges, that is:

$$D \triangleq \{(v,e) \mid v \in e,\ e = \{v_1, \ldots, v_k\} \text{is an hyperegde}\} \tag{1}$$

Since $\bar{d} = |D|/n$, we show how to approximate $|D|$. For clarity reasons, we shall assume for now that there are no disconnected vertices (i.e. $\forall v \in V, d(v) \geq 1$).

The basic idea of the algorithm is to sample vertices and to put them into "buckets" according to their degrees such that in bucket B_i we have vertices with degree between $(1+\beta)^{i-1}$ and $(1+\beta)^i$ (where $\beta = \varepsilon/c$ for some constant $c > 1$). If S is the sample, then we denote by S_i the subset of sampled vertices that belong to B_i. We will focus on the sets S_i that are *sufficiently large*, because we want $|S_i|/|S|$ to be a good approximation of $|B_i|/n$. Let us denote the set of the such i's by L.

Indeed, the sum $(n/|S|)\sum_{i\in L}|S_i|(1+\beta)^{i-1}$ approximates well the number of directed hyperedges outgoing from a vertex that resides in such a large bucket. Taking a sample size $|S|$ of an adequate size ensures that with high probability this sum approximates well $\sum_{i\in L}|B_i|(1+\beta)^{i-1} \leq |D|$. In this case, the approximating sum overestimates $|D|$ by a factor of at most $(1+\varepsilon)$.

Although using this basic approximation yields an overestimating factor of at most $(1+\varepsilon)$, it may underestimate $|D|$ by a factor of k. As we shall show next, this obtains a basic $(k+\varepsilon)$-approximation that corresponds to the basic $(2+\varepsilon)$-approximation presented by Goldreich and Ron (see [2]).

Let us focus now on the directed hyperedges that are ignored by the basic algorithm. They are all outgoing hyperedges from a vertex that resides in a small bucket. As such, approximations of their size are unreliable. We denote the set of vertices that reside in any such small bucket by

$$U \triangleq \cup_{i\notin L} B_i \tag{2}$$

Note, that any underestimation below a $(1+\varepsilon)$ factor that the basic algorithm makes, is due to those directed hyperedges outgoing from U. There are two basic types of such directed hyperedges: the first is due to hyperedges that consist solely of vertices in U (i.e. those hyperedges for which $e \subset U$), and the second is those directed hyperedges that have vertices both in U and in $\cup_{i\in L} B_i$. We refer to these as "mixed" hyperedges.

As for directed hyperedges of the first "unmixed" type, their number is upper bounded by $|U|^k$. This calls for setting the largeness threshold of the buckets to approximately $(\varepsilon n)^{\frac{1}{k}}$. Accordingly, the number of directed hyperedges of this type is at most $O(\varepsilon n)$ (recall that $|D| \geq n$), and therefore they may be neglected. This requires a sample of size $\tilde{O}(poly(\frac{1}{\varepsilon}) \cdot n^{\frac{k-1}{k}})$ in order to approximate well the number of vertices residing in the large buckets.

It follows that the source of the inaccuracy is only due to the directed hyperedges of the second ("mixed") type. These are directed hyperedges outgoing from U that incident both at U and at $V \setminus U = \cup_{i \in L} B_i$. Note also, that their corresponding directed hyperedges outgoing from $V \setminus U$ were already counted in their appropriate buckets. This type of hyperedges may have m vertices in U and the other $k - m$ vertices in $V \setminus U$, for $m \in \{1, \ldots, k-1\}$. In the general case, only those directed hyperedges in $V \setminus U$ were counted $k - m$ times, whereas the m directed hyperedges outgoing from vertices in U were ignored. In the (worst) case where $m = k - 1$, only one directed hyperedge was counted, ignoring the other $k - 1$ directed hyperedges. This asserts that the basic algorithm underestimates D by a factor of at most k, and establishes the $(k + \varepsilon)$-approximation.

Note that the basic algorithm uses only degree queries. We show now that using only such queries, a $k + \varepsilon$ is the best approximation possible. As an example for such underestimation, the reader may think of a (small) set of $k-1$ vertices, where the hyperedges are all the subsets of size k that include those $k-1$ vertices. The degree of almost every vertex is 1, whereas $|D|$ is roughly kn. Compare this case with a hypergraph where the hyperedges are a partition of the graph, each of size k. In the latter case, all the degrees are 1. It follows, that any algorithm that makes only $o(n)$ degree queries will see only 1s, and cannot distinguish the two hypergraphs. Therefore it can approximate $|D|$ by a factor of at most k.

We shall now explain how to use neighbour queries in order to overcome the k barrier and achieve a $(1 + \varepsilon)$-approximation to $|D|$. It is left to show how to approximate the "mixed" directed hyperedges outgoing from U, which have vertices both in U and $V \setminus U$. Recall that these were ignored by the basic algorithm, and were not counted. We cannot estimate these based on quantities sampled directly from U, since these are unreliable. Instead, we show the connection between them and the corresponding hyperedges outgoing from $V \setminus U$, and sample the latter accordingly.

Let us introduce a piece of notation. For two disjoint sets of vertices V_1 and V_2, and positive integer numbers l and m such that $l + m \leq k$, we denote the set of directed hyperedges outgoing from V_1 that have exactly m vertices in V_1 and l vertices in V_2 by

$$D_{(m,l)}(V_1, V_2) \triangleq \{(v, e) | (v, e) \in D, v \in V_1, |e \cap V_1| = m, |e \cap V_2| = l\} \tag{3}$$

Whenever the sets V_1 and V_2 form a partition of V (i.e. $V_1 \dot{\bigcup} V_2 = V$), the set may be non-empty only if $l = k - m$.

Under these notations, the "mixed" directed hyperedges outgoing from U with m vertices in U (and in turn $k - m$ vertices in $V \setminus U = \cup_{i \in L} B_i$), are exactly the directed hyperedges in the set

$$D_{(m,k-m)}(U, \cup_{i\in L} B_i) \tag{4}$$

Since each such (undirected) hyperedge (with m vertices in U and $k-m$ vertices in $\cup_{i\in L} B_i$) is counted as m directed hyperedges in U and $k-m$ directed hyperedges in $\cup_{i\in L} B_i$, it holds that:

$$\frac{|D_{(m,k-m)}(U, \cup_{i\in L} B_i)|}{m} = \frac{|D_{(k-m,m)}(\cup_{i\in L} B_i, U)|}{k-m} \tag{5}$$

Consider now a directed hyperedge of the form (v, e) in the set $D_{(k-m,m)}(\cup_{i\in L} B_i, U)$. It must be that $v \in B_i$ for some $i \in L$. Since e has exactly m vertices in U, it has at most $k-m$ vertices in that B_i. Therefore, it resides in $D_{(l,m)}(B_i, U)$ for some $l \leq k-m$. Accordingly, consider a directed hyperedge in a set of the form $D_{(l,m)}(B_i, U)$ for some $i \in L$ and $l \leq k-m$. Since it has m vertices in U, it must also reside in the set $D_{(k-m,m)}(\cup_{i\in L} B_i, U)$. It follows that:

$$D_{(k-m,m)}(\cup_{i\in L} B_i, U) = \dot{\bigcup_{i\in L}} \dot{\bigcup_{l=1}^{k-m}} D_{(l,m)}(B_i, U) \tag{6}$$

Using Eq. (5) and Eq. (6), we have that the number of directed hyperedges outgoing from U and having at least one vertex in $V \setminus U$ is:

$$\begin{aligned}
\sum_{m=1}^{k-1} |D_{(m,k-m)}(U, \cup_{i\in L} B_i)| &= \sum_{m=1}^{k-1} \frac{m}{k-m} \sum_{i\in L} \sum_{l=1}^{k-m} \left|D_{(l,m)}(B_i, U)\right| &(7)\\
&= \sum_{i\in L} \sum_{m=1}^{k-1} \sum_{l=1}^{k-m} \frac{m}{k-m} \left|D_{(l,m)}(B_i, U)\right| &(8)
\end{aligned}$$

Therefore, it remains to approximate the sets $D_{(l,m)}(B_i, U)$ for $i \in L$. Note that such a set is a subset of the set of directed hyperedges outgoing from B_i denoted by.

$$D_i \triangleq \{(v, e) | v \in B_i, (v, e) \in D\} \tag{9}$$

We define the density of such a set $D_{(l,m)}(B_i, U)$ of directed hyperedges among the directed hyperedges in D_i to be

$$\alpha_{l,m}(i) \triangleq \frac{\left|D_{(l,m)}(B_i, U)\right|}{|D_i|} \tag{10}$$

It follows that $D_{(l,m)}(B_i, U) = \alpha_{l,m}(i) \cdot |D_i|$. We already have a good approxiamtion of $|D_i|$ by $\frac{n \cdot |S_i|}{|S|}(1+\beta)^{i-1}$. Therefore, a $(1+\varepsilon)$-approximation of $\alpha_{l,m}(i)$ will establish a $(1+\varepsilon)$-approximation of $D_{(l,m)}(B_i, U)$.

Since all the degrees of the vertices in such a bucket are almost the same, the distribution we get by sampling a random hyperedge for a vertex v in S_i is close to the uniform distribution on D_i. For each of the sampled vertices in S_i, we pick a random hyperedge and check wheather it's in $D_{(l,m)}(B_i, U)$. We use the proportion of the sampled directed hyperedges as our estimation for $\alpha_{l,m}(i)$.

These quantites complete the approximation of $D_{(l,m)}(B_i, U)$, and establish the $(1+\varepsilon)$-approximation of $|D|$. We now present and analyze the algorithm.

2.3 The Algorithm and its Analysis

The algorithm we give approximates $|D|$. It is given the approximation parameter $\varepsilon \leq \frac{1}{2}$, and an a priori lower bound, ℓ, on $\bar{d}$. We later explain how to get rid of ℓ. The reader may think of $\ell = 1$, as in the foregoing motivating discussion.

We set $\beta \triangleq \varepsilon/8$ and $t \triangleq \lceil \log_{(1+\beta)} n^k \rceil + 1$. Let us define a partition of V into the following t buckets:

$$B_i = \left\{ v : \ d(v) \in \left((1+\beta)^{i-1}, (1+\beta)^i \right] \right\}, \quad \text{for } i = 0, 1, \ldots, t-1 \ . \tag{11}$$

The algorithm works as follows:

The Approximation Algorithm

1. Uniformly and independently select $s = \tilde{O}(k^3 \cdot \varepsilon^{-(3+\frac{1}{k})} \cdot \ell^{(-1/k)} \cdot n^{\frac{k-1}{k}})$ vertices from V, and let S denote the (multi-)set of selected vertices.

2. Set $S_i \triangleq S \cap B_i$

3. Define $L \triangleq \{i : \frac{n \cdot |S_i|}{|S|} > \frac{1}{t} \cdot (\frac{\varepsilon}{6} \cdot n \cdot \ell)^{\frac{1}{k}}\}$

4. For every $i \in L$ and $v \in S_i$:

   (a) pick a random hyperedge of v.

   (b) Query the degrees of the members of the hyperedge.

(c) Let $\chi_{(l,m)}(v) = 1$ if the hyperedge that was chosen has exactly l vertices in B_i, and m vertices in U, and $\chi_{(l,m)}(v) = 0$ otherwise.

5. For $i \in L$, set $\tilde{\alpha}_{(l,m)}(i) = \frac{|v \in S_i : \chi_{(l,m)}(v)=1|}{|S_i|}$

6. Output $A \triangleq \sum_{i \in L} \frac{n \cdot |S_i|}{|S|}(1 + \sum_{m=1}^{k-1} \sum_{l=1}^{k-m} \frac{m}{k-m} \tilde{\alpha}_{(l,m)}(i))$

We show now the correctness of the algorithm:

Lemma 1. *The output value A of the algorithm satisfies $(1-\varepsilon)|D| < A < (1+\varepsilon)|D|$, with probability at least $2/3$.*

Proof. By the definition of the buckets (See Eq. (11)) it holds that:

$$|D| \;\leq\; \frac{1}{n}\sum_{i=1}^{t} |B_i| \cdot (1+\beta)^i \;\leq\; (1+\beta) \cdot |D| \tag{12}$$

The sample size s is chosen such that it will provide a good approximation to the big buckets according to a largeness threshold of $\psi \triangleq \frac{1}{t} \cdot (\frac{\varepsilon}{8} \cdot \ell \cdot n)^{\frac{1}{k}}$ vertices. Note that this is a slightly smaller threshold than the one that the algorithm uses, in order to assure the actual bound on the size of the set U (recall that $U = \cup_{i \notin L} B_i$). It follows by using the multiplicative Chernoff Bound that, with high probability, it holds that:

$$\forall i \;\text{ s.t. }\; |B_i| \geq \psi : \quad \left(1 - \frac{\varepsilon}{4k}\right) \cdot |B_i| \;\leq\; \frac{|S_i| \cdot n}{|S|} \;\leq \left(1 + \frac{\varepsilon}{4k}\right) \cdot |B_i| \tag{13}$$

and

$$\forall i \;\text{ s.t. }\; |B_i| < \psi : \quad \frac{|S_i| \cdot n}{|S|} < \frac{1}{t} \cdot \left(\frac{\varepsilon}{6} \cdot \ell \cdot n\right)^{\frac{1}{k}} \tag{14}$$

By (14), if $|B_i| < \psi$ then $i \notin L$. Yet, it may be the case that a bucket that is well approximated (i.e. $|B_i| > \phi$) is placed into L. However, we show that in this case, its size is at most $\frac{1}{t} \cdot (\frac{\varepsilon}{4} \cdot \ell \cdot n)^{\frac{1}{k}}$. By the left side of Eq. (13), it follows that:

$$\left(1 - \frac{\varepsilon}{4k}\right) \cdot \frac{1}{t} \cdot \left(\frac{\varepsilon}{4} \cdot \ell \cdot n\right)^{\frac{1}{k}} > \frac{1}{t} \cdot \left(\frac{\varepsilon}{6} \cdot \ell \cdot n\right)^{\frac{1}{k}} \tag{15}$$

iff

$$\left(1 - \frac{\varepsilon}{4k}\right)^{k} > \frac{4}{6} \tag{16}$$

which holds for $k \geq 2$ and $\varepsilon < \frac{1}{2}$.

Thus, we have that $|U| < (\frac{\varepsilon}{4} \cdot \ell \cdot n)^{\frac{1}{k}}$. It follows that the total number of directed hyperedges that incident only between the vertices of the set U is bounded by:

$$|U|^k \leq \frac{\varepsilon}{4} \cdot n \cdot \ell < \frac{\varepsilon}{4}|D| \tag{17}$$

Following the analysis in the previous section, this verifies that:

$$(1 - \frac{\varepsilon}{4})|D| < \sum_{i \in L} |D_i| \cdot \left(1 + \sum_{m=1}^{k-1} \sum_{l=1}^{k-m} \frac{m}{k-m} \cdot \alpha_{(l,m)}(i)\right) < |D| \tag{18}$$

We now analyze the estimation of the proportions $\alpha_{(l,m)}(i)$ by $\tilde{\alpha}_{(l,m)}(i)$ for $i \in L$. We next show that for every $i \in L$ and every $\alpha_{(l,m)}(i) \geq \frac{\varepsilon}{8k^3}$, it holds that:

$$\left(1 - \frac{\varepsilon}{4}\right) \cdot \alpha_{(l,m)}(i) \ \leq \ \tilde{\alpha}_{(l,m)}(i) \ \leq \ \left(1 + \frac{\varepsilon}{4}\right) \cdot \alpha_{(l,m)}(i) \tag{19}$$

and for every $\alpha_{(l,m)}(i) < \frac{\varepsilon}{8k^3}$ it holds that:

$$\alpha_{(l,m)}(i) < \frac{\varepsilon}{4k^3} \tag{20}$$

Assuming that the estimations of the size of the buckets B_i so far are good, we have $\Omega(\frac{k^3}{\varepsilon^3})$ hyperedges sampled from each bucket B_i for $i \in L$. Thus, the multiplicative Chernoff Bound establishes that both Eq. (19) and Eq. (20) hold with high probability.

Note that our algorithm samples the hyperedges of D_i in an almost uniform manner. In fact, when restricting the selections of vertices of the algorithm to the set S_i, the algorithm first samples a vertex of B_i and then samples at random one of its hyperedges. By this, the proportion sampled (say $\tilde{\tilde{\alpha}}_{(l,m)}(i)$) differs by at most a $(1+\beta)$ factor from the real proportion of the directed hyperedges of the set $D_{(l,m)}(B_i, U)$ in D_i. Thus, approximating $\tilde{\tilde{\alpha}}_{(l,m)}(i)$ with a multiplicative error of a $(1+\varepsilon/8)$ factor establishes Eq. (19).

In this case, the following holds:

$$
\begin{aligned}
A &= \sum_{i\in L} \frac{n\cdot|S_i|}{|S|}\left(1+\sum_{m=1}^{k-1}\sum_{l=1}^{k-m}\frac{m}{k-m}\tilde{\alpha}_{(l,m)}(i)\right) && (21)\\
&> (1-\frac{\varepsilon}{4})\sum_{i\in L}|D_i|\left(1+\sum_{m=1}^{k-1}\sum_{l=1}^{k-m}\frac{m}{k-m}\tilde{\alpha}_{(l,m)}(i)\right) && (22)\\
&\geq (1-\frac{\varepsilon}{4})\sum_{i\in L}|D_i|\Bigg(1+(1-\frac{\varepsilon}{4})\sum_{m=1}^{k-1}\sum_{l=1}^{k-m}\frac{m}{k-m}\cdot\alpha_{(l,m)}(i) \\
&\qquad - \sum_{m=1}^{k-1}\sum_{l=1}^{k-m}\frac{m}{k-m}\cdot\frac{\varepsilon}{4k^3}\Bigg) && (23)\\
&\geq (1-\frac{\varepsilon}{4})\sum_{i\in L}|D_i|\left((1-\frac{\varepsilon}{4})+(1-\frac{\varepsilon}{4})\sum_{m=1}^{k-1}\sum_{l=1}^{k-m}\frac{m}{k-m}\cdot\alpha_{(l,m)}(i)\right) && (24)\\
&= (1-\frac{\varepsilon}{4})^2\sum_{i\in L}|D_i|\left(1+\sum_{m=1}^{k-1}\sum_{l=1}^{k-m}\frac{m}{k-m}\cdot\alpha_{(l,m)}(i)\right) && (25)\\
&> (1-\frac{\varepsilon}{4})^3\cdot|D| \geq (1-\varepsilon)\cdot|D| && (26)
\end{aligned}
$$

In Eq. (22) and Eq. (23), we use the estimations on the size of the buckets and the proportions of the directed hyperedges outgoing from them. The last inequality, uses the bound on $|D|$ presented in Eq. (18). Similarily, we have that $A < (1+\varepsilon)\cdot|D|$.

The correctness of the lemma follows. □

3 On approximating the sum of distances between all pairs of points

3.1 Introduction

We consider a given set $S \triangleq \{P_1, \ldots, P_n\}$ of n points in R^d, where $P_i = (p_{i,1}, \ldots, p_{i,d})$. For every pair of points (P_i,P_j) , we denote by δ_{ij} their Euclidean distance, that is $\delta_{ij} = \|P_i - P_j\|_2 = (\sum_{k=1}^{d} (p_{i,k} - p_{j,k})^2)^{1/2}$. The sum of distances between all the pairs of points in this set is $\mathbb{S}(S) = \sum_{i \neq j} d_{ij}$. The trivial algorithm for calculating this sum (or equivalently the average distance), which simply sums all the pairs of points, runs in time $O(dn^2)$.

We note that for the case in which all the points are arranged on the line (i.e., $d = 1$), there is a known algorithm which runs in time $O(n \log n)$. We begin by surveying the case $d = 1$, since it shall be used as a black-box for the higher dimension estimations.

We then present a deterministic $\sqrt{d}$-approximation that always yields a correct approximation. The algorithm is based on the observation that, by using the 1-dimensional case on each of the d axes, we can compute the sum of the *norm-1* distances between all pairs of points. Using the relation between norms, this gives a deterministic $\sqrt{d}$-approximation of the sum in time $O(dn \log n)$. Moreover, we never underestimate the sum.

The idea for the our main algorithm is that instead of using the standard basis as its coordinate system as done in the $\sqrt{d}$-approximation, we use a random coordinate system. It follows by linearity of expectation (applied to the sum of all distances), that the expected value of the sum of projected points equals $f(d)$ times the correct sum, where $f(d) \approx \frac{1}{\sqrt{d}}$ has a closed form. (Note that $f(d)$ is the expected length of the projection of a unit d-dimensional vector in a random direction.)

The main result In this section we show a randomized $(1 + \varepsilon)$-approximation of $\mathbb{S}(S)$ having running time of $O(\frac{dn \log n}{\varepsilon^2})$.

3.2 The case of a Line ($d = 1$)

In the case where all points are located on the line, we denote the set of points $S = (p_1, \ldots, p_n)$. The known algorithm exploits the fact that points on a line are comparable (i.e. for every x, y either $x > y$ or $x \leq y$), and for $x < y < z$ it holds that $\|x - z\| = \|x - y\| + \|y - z\|$. For a given point p_i, we denote $S_i = \sum_{j=1}^{n} |p_i - p_j|$ and we may present the sum of all the distances as follows:

$$\mathbb{S}(S) = \sum_{i \neq j} |p_i - p_j| = \sum_{i=1}^{n} S_i \quad . \tag{27}$$

Each S_i may be computed in $O(n)$ time. However, when the points of S are sorted (say in a non-decreasing order, i.e., $p_1 \leq p_2 \leq \cdots \leq p_n$), for every $i \in \{1, \ldots, n-1\}$ it holds that

$$S_{i+1} = S_i + (2i - n) \cdot (p_{i+1} - p_i) \quad . \tag{28}$$

This is proven by the straightforward manipulation:

$$\begin{aligned}
S_{i+1} &= \sum_{j=1}^{n} |p_{i+1} - p_j| \\
&= \sum_{j \leq i} (p_{i+1} - p_j) + \sum_{j \geq i+2} (p_j - p_{i+1}) \\
&= \sum_{j \leq i-1} (p_{i+1} - p_i + p_i - p_j) + (p_{i+1} - p_i) + \sum_{j \geq i+2} (p_j - p_i - (p_{i+1} - p_i)) \\
&= \sum_{j=1}^{n} |p_i - p_j| + [(i-1) - (n-i-1)] \cdot (p_{i+1} - p_i) \\
&= S_i + (2i - n) \cdot (p_{i+1} - p_i)
\end{aligned}$$

This suggests an algorithm that works in sub-quadratic time, as follows: Given the multiset S, we first sort the points. We shall now assume that the points of S are sorted in a non-decreasing order, i.e., $p_1 \leq \ldots \leq p_n$. We compute S_1 by summing the distances $(p_j - p_1)$ for $j = 2, \ldots, n$. We initialize $K \triangleq S_1$, and for $i = 1, \ldots, n-1$, we compute $S_{i+1} = S_i + (2i - n)(p_{i+1} - p_i)$ and update $K \leftarrow K + S_{i+1}$. The Algorithm returns K as its output.

Equality (28) shows that the algorithm computes at each stage the correct S_{i+1} based on S_i, and equality (27) asserts that the returned value is in fact the sum of all distances between the points of S. As for the running time, sorting the points is done in time $O(n \log n)$, calculating S_1 is done in time $O(n)$, each of the $n-1$ sums of the form S_i is calculated in constant time, which yields a total running time of $O(n \log n)$.

3.3 Deterministic $\sqrt{d}$-approximation for $\mathbb{S}(S)$ in R^d

We now present a deterministic algorithm that achieves a $\sqrt{d}$-approximation for $\mathbb{S}(S)$ and works in time $O(dn \log n)$. Moreover, the algorithm never underestimates $\mathbb{S}(S)$. Our algorithm utilizes the deterministic algorithm for the line and a well-known norms inequality.

The *norm-1* for a point $\bar{X} = (x_1, \ldots, x_d)$ is defined as $\sum_{k=1}^{d} |x_k|$. Accordingly, for a pair of points, $\bar{P}_i$ and $\bar{P}_j$ their the norm-1 distance is $\sum_{k=1}^{d} |P_{i,k} - P_{j,k}|$.

Therefore, the sum of norm-1 distances between all pairs of points is given by:

$$\sum_{i \neq j} \|P_i - P_j\|_1 = \sum_{i \neq j} \sum_{k=1}^{d} |P_{i,k} - P_{j,k}| = \sum_{k=1}^{d} \left(\sum_{i \neq j} |P_{i,k} - P_{j,k}| \right) \tag{29}$$

The key obeservation is that the sum in parentheses on the right side of (29) is just the sums of distances of the projection of the points on the corresponding axes. Thus, we may computer (29) by invoking the line algorithm d times.

For any $\bar{X}$ in R^d, the following norms inequality holds:

$$\|\bar{X}\|_2 \leq \|\bar{X}\|_1 \leq \sqrt{d}\|\bar{X}\|_2 \tag{30}$$

Applying Eq. (30) on the sum of all distances, we get:

$$\sum_{i \neq j} \|P_i - P_j\|_2 \leq \sum_{i \neq j} \|P_i - P_j\|_1 \leq \sqrt{d} \sum_{i \neq j} \|P_i - P_j\|_2 \tag{31}$$

To conclude, the algorithm sums the outputs of d 1-dimensional problems, where the input to the i'th problem is $\{P_{1,i}, \ldots, P_{n,i}\}$. Inequality (31) establishes the desired approximation. The d iterations of the 1-dimensional algorithm run in time $O(dn \log n)$.

3.4 Projection on a random direction

We begin by observing that the $\sqrt{d}$-approximation derived in Section 3.3 can be based on any (orthogonal) coordinate system. The choice of the standard basis in the description was in fact arbitrary. Motivated by this method, we examine the expected value of the sum of all distances in *norm-1* with respect to a *random coordinate system*. We will show that this expectation equals d times the expected value the sums of all distances between all projected points in a single direction. This suggests to estimate $\mathbb{S}(S)$ by projecting the points on a single random direction. In turn, we will show that the expectation of a projection on a single random direction equals $f(d)$ times the sum of the norm-2 distances, where f is a function of d.

Let $\vec{e_1}, \ldots, \vec{e_d}$ be such a random orthonormal coordinate system. The sum of all distances w.r.t. *norm-1* induced by the system is:

$$\sum_{i \neq j} \sum_{k=1}^{d} \langle \vec{e_k}, P_i - P_j \rangle = \sum_{k=1}^{d} \sum_{i \neq j} \langle \vec{e_k}, P_i - P_j \rangle \tag{32}$$

Using linearity of Expectation and the fact that axes are randomly chosen, we have that the expectation of this random sum is d times the expectation of the sum of the distances of the projection of all points on a single random direction. This is shown by the following manipulation:

$$\mathbb{E}_{\vec{e_1},\ldots,\vec{e_d}} \left[\sum_{k=1}^{d} \sum_{i \neq j} \langle \vec{e_k}, P_i - P_j \rangle \right] = \sum_{k=1}^{d} \mathbb{E}_{\vec{e_k}} \left[\sum_{i \neq j} \langle \vec{e_k}, P_i - P_j \rangle \right] = d \cdot \mathbb{E}_{\vec{r}} \left[\sum_{i \neq j} \langle \vec{r}, P_i - P_j \rangle \right]$$

where $\vec{r}$ is a random direction vector. Thus, we shall further on focus on a projection on a single random direction.

In fact, it is a property of the uniform distribution on the unit sphere that for any unit vector $\vec{v} : \|\vec{v}\|_2 = 1$, the expected value of the random variable $|\langle \vec{v}, \vec{r} \rangle|$ does not depend on the direction of v. As shown in Section 3.6, this value depends only on the dimension d. We denote it by:

$$f(d) \triangleq \mathbb{E}_{\vec{r}}[|\langle \vec{r}, \vec{v} \rangle|] \tag{33}$$

For every pair of points in the d-dimensional space, P_i, P_j, we denote their distance vector as:

$$D_{ij} \triangleq P_i - P_j = \delta_{ij} \cdot \vec{\omega_{ij}} \tag{34}$$

where $\delta_{ij} = \|D_{ij}\|_2$ is its magnitude, and $\vec{\omega_{ij}} = \frac{D_{ij}}{\delta_{ij}}$ is its direction vector.

Under a projection on a specific vector $\vec{r}$, it holds that the distance between the projected points is $\delta_{ij} \cdot |\langle \vec{r}, \vec{\omega_{ij}} \rangle|$, since a projection is a linear transformation.

Taking $\vec{r}$ to be a random direction chosen from the uniform distribution on the unit sphere, we define:

$$Y_{ij} \triangleq |\langle \vec{r}, \vec{\omega_{ij}} \rangle| \tag{35}$$

as the random length of $\vec{\omega_{ij}}$ projected along $\vec{r}$.

In addition, we define the random variable Z to be the random sum of all distances between the pairs of the projected points along the random direction $\vec{r}$. That is:

$$Z \triangleq \sum_{i \neq j} \delta_{ij} Y_{ij} \tag{36}$$

By linearity of expectation, we get:

$$\mathbb{E}[Z] = \sum_{i \neq j} \delta_{ij} \mathbb{E}[Y_{ij}] \tag{37}$$

Combining the definition of the Y_{ij} variables with Eq. (33), we get:

$$\mathbb{E}[Z] = \sum_{i \neq j} \delta_{ij} \mathbb{E}\left[\langle \vec{r}, \vec{\omega_{ij}} \rangle\right] = \sum_{i \neq j} f(d) \cdot \delta_{ij} = f(d) \cdot \mathbb{S}(S) \tag{38}$$

We conclude this section with a simple bound on the variance of Z. The variance of Z equals $\mathbb{E}[Z^2] - (\mathbb{E}[Z])^2$, and therefore is always upper-bounded by $E[Z^2]$. Using the fact that all the Y_{ij} are positive and are at most 1, we have:

$$\mathbb{E}[Z^2] = \mathbb{E}[(\sum_{i \neq j} \delta_{ij} Y_{ij})^2] \leq \mathbb{E}[\sum_{i \neq j} (\delta_{ij})^2] = (\mathbb{S}(S))^2$$

A more careful bound of the variance of Z, by $O(\frac{1}{d} \cdot \mathbb{S}(S)^2)$, is described in Section 3.7.

3.5 A $(1+\varepsilon)$-approximation for $\mathbb{S}(S)$ in R^d

The analysis of a single projection on a random direction suggests the following algorithm:

Pick a sample of K independent random directions. For each direction project the set of points on it, and calculate the sum of distances between all the pairs of the projected points using the 1-dimensional case. Output the average over the samples multiplied by $\frac{1}{f(d)}$.

A standard pairwise independent sampling scheme yields the correctness of the algorithm for $K = \Theta(\frac{1}{\varepsilon^2})$. Denote by Z_i the sum of the distances calculated in the i^{th} projection, where the Z_is are distributed as Z and are pairwise independent. Recall that $\mathbb{E}[\frac{Z_i}{f(d)}] = \mathbb{S}(S)$. Using the Chebyschev Inequality we get that the probability of the event that the output $(\frac{\sum_{i=1}^{K} Z_i}{K \cdot f(d)})$ is not a $(1+\varepsilon)$-approximation of $\mathbb{S}(S)$ is:

$$Pr\left[\left|\frac{\sum_{i=1}^{K} Z_i}{K} - f(d)\cdot\mathbb{S}(S)\right| > \varepsilon\cdot f(d)\cdot\mathbb{S}(S)\right] < \frac{Var(Z)}{\varepsilon^2\cdot f(d)^2\cdot\mathbb{S}(S)^2\cdot K}$$

Recalling that $Var(Z) = O(\frac{1}{d}\cdot(\mathbb{S}(S))^2)$ and $f(d) = \Theta(\frac{1}{\sqrt{d}})$, it suffices to make $K = \Theta(\frac{1}{\varepsilon^2})$ such samples in order to reduce the error probability below $\frac{1}{3}$.

The time complexity of the suggested algorithm is $\Theta(\frac{dn+n\log n}{\varepsilon^2}) = O(\frac{dn\log n}{\varepsilon^2})$, because a single projection is done in time dn, and the 1-dimensional case algorithm works in time $n\log n$, for each of the K repetitions.

3.6 The Notorious $f(d)$

We now return to the issue of calculating the expected length of a projection of a randomly chosen unit sphere vector on a fixed unit sphere vector. We shall show that this value (denoted $f(d)$) is in fact $\Theta(\frac{1}{\sqrt{d}})$.

We first note that this value does not depend on the direction of a specific vector, but only on d, its dimension. This follows directly from the fact that a random projection of the uniform distribution on the sphere is invariant under rotations (see Appendix A). We therefore assume that we project to the d^{th} unit vector of the standard basis, $\vec{e_d} = (0,\ldots,0,1)^\tau$. Thus, for a random vector $\vec{r}$ on the sphere, we seek the value of $\mathbb{E}_{\vec{r}}\left[|\langle\vec{r},\vec{e_d}\rangle|\right]$.

In general, for every d-dimensional distribution X with density function f_x, and every $g : R^d \to R$ continuous function, the expectation of $g(X)$ is given by $\int_{R^d} g(X)f_x(\vec{x})\partial\vec{x}$.

The uniform distribution on the sphere in R^d is generated by taking the direction vector of a d-dimensional normally distributed vector (see Appendix A). We may use its density function to calculate the required expected value. That is, $\vec{r} = \frac{\vec{x}}{\|\vec{x}\|_2}$, where $\vec{x}$ is distributed according to the d-dimensional normal distribution. Recall that we denote this expectation by $f(d)$, that is:

$$f(d) \triangleq \mathbb{E}_{\vec{r}}\left[|\langle\vec{r},\vec{e_d}\rangle|\right] = \mathbb{E}_{\vec{x}}\left[\left|\left\langle\frac{\vec{x}}{\|\vec{x}\|_2},\vec{e_d}\right\rangle\right|\right]$$

$$= \int_{\mathbb{R}}\cdots\int_{\mathbb{R}} \frac{1}{(2\pi)^{d/2}}\left|\left\langle\frac{\vec{x}}{\|\vec{x}\|_2},\vec{e_d}\right\rangle\right| e^{\frac{-(\vec{x},\vec{x})}{2}}\partial\vec{x} \tag{39}$$

We present our calculation of the integral in (39), using a change of variables. Representing the integral in spherical coordinates allow us to calculate its value by comparison to the normal distribution.

The transformation $T : [0, \infty) \times [0, \pi) \times \cdots \times [0, \pi) \times [0, 2\pi) \to R^d$, defined by

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{d-2} \\ x_{d-1} \\ x_d \end{pmatrix} = T \begin{pmatrix} \rho \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_{d-2} \\ \theta_{d-1} \end{pmatrix} \triangleq \begin{pmatrix} \rho \sin(\theta_1) \sin(\theta_2) \ldots \sin(\theta_{d-2}) \sin(\theta_{d-1}) \\ \rho \sin(\theta_1) \sin(\theta_2) \ldots \sin(\theta_{d-2}) \cos(\theta_{d-1}) \\ \vdots \\ \rho \sin(\theta_1) \sin(\theta_2) \cos(\theta_3) \\ \rho \sin(\theta_1) \cos(\theta_2) \\ \rho \cos(\theta_1) \end{pmatrix}$$

where $\rho \in (0, \infty)$ denotes the length of $\bar{x}$ and $\theta_1, \ldots, \theta_{d-2} \in [0, \pi)$ and $\theta_{d-1} \in [0, 2\pi)$ are angles that determine its direction.

The Jacobian of this transformation, is known to satisfy:

$$|J_T| = \rho^{d-1} \sin^{d-2}(\theta_1) \sin^{d-3}(\theta_2) \ldots \sin^2(\theta_{d-3}) \sin(\theta_{d-2}) \tag{40}$$

Under spherical coordinates, the term $\langle \frac{\vec{x}}{\|\vec{x}\|_2}, \vec{e_d} \rangle$ equals $\frac{\rho \cos\theta_1}{\rho} = \cos\theta_1$. Thus, applying the change of variables induced by the transformation T^{-1}, we get:

$$\begin{aligned} f(d) &= \int\limits_{\mathbb{R}^+} \int\limits_{[0,\pi)} \cdots \int\limits_{[0,\pi)} \int\limits_{[0,2\pi)} \frac{1}{(2\pi)^{d/2}} |\cos\theta_1| \, e^{\frac{-\rho^2}{2}} |J_T| \partial\theta_{d-1} \ldots \partial\theta_1 \partial\rho \qquad (41) \\ &= \int\limits_{[0,\pi)} |\cos\theta_1| \sin^{d-2}(\theta_1) \partial\theta_1 \cdot \left(\int\limits_{\mathbb{R}^+} \int\limits_{[0,\pi)} \cdots \int\limits_{[0,\pi)} \int\limits_{[0,2\pi)} \frac{1}{(2\pi)^{d/2}} e^{\frac{-\rho^2}{2}} \rho^{d-1} \prod_{j=2}^{d-2} \sin^{d-1-j}\theta_j \partial\theta_{d-1} \ldots \partial\theta_2 \partial\rho \right) \end{aligned}$$

We now exploit the fact that the normal distribution is a probability mass. As such, the integral of its density function over R^d equals 1. Applying the same change of coordinates, we have:

$$1 = \int\limits_{\mathbb{R}} \dots \int\limits_{\mathbb{R}} \frac{1}{(2\pi)^{d/2}} e^{\frac{-(\vec{y},\vec{y})}{2}} \partial\vec{y} \tag{42}$$

$$= \int\limits_{[0,\pi)} \sin^{d-2}(\theta_1)\partial\theta_1 \cdot \left(\int\limits_{\mathbb{R}^+} \int\limits_{[0,\pi)} \dots \int\limits_{[0,\pi)} \int\limits_{[0,2\pi)} \frac{1}{(2\pi)^{d/2}} e^{\frac{-\rho^2}{2}} \rho^{d-1} \prod_{j=2}^{d-2} \sin^{d-1-j}\theta_j \partial\theta_{d-1} \dots \partial\theta_2 \partial\rho \right)$$

Denoting $A_k \triangleq \int\limits_{[0,\frac{\pi}{2}]} \sin^k\theta\partial\theta$, and dividing (41) by (42), we get:

$$f(d) = \frac{\int\limits_{[0,\frac{\pi}{2}]} \cos\theta_1 \sin^{d-2}\theta_1 \partial\theta_1}{A_{d-2}} \tag{43}$$

The numerator evaluates to $\frac{1}{d-1}$ by a simple change of variables, whereas integration by parts of A_k, yields the recursive formula:

$$A_k = \frac{k-1}{k} A_{k-2} \tag{44}$$

Along with the starting conditions that $A_0 = \frac{\pi}{2}$ and $A_1 = 1$, we have:

$$\begin{aligned} f(d) &= \frac{1}{d-1} \cdot \frac{1}{A_{d-2}} = \begin{cases} \dfrac{\prod\limits_{i=1}^{\frac{d-3}{2}} 2i+1}{\prod\limits_{i=1}^{\frac{d-1}{2}} 2i} & , d \text{ is odd} \\ \dfrac{\prod\limits_{i=1}^{\frac{d-2}{2}} 2i}{\prod\limits_{i=1}^{\frac{d-2}{2}} 2i+1} \cdot \frac{2}{\pi} & , d \text{ is even} \end{cases} \\ &= \Theta(\frac{1}{\sqrt{d}}) \end{aligned}$$

The Wallis' inequality (see [5]) asserts the bound. Specifically, we get $f(2) = \frac{2}{\pi}$ and $f(3) = \frac{1}{2}$.

3.7 Analyzing the Variance of the random sum Z

Now we provide a tighter bound for the variance of the projected sum Z. Recall that a bound on $\mathbb{E}[Z^2]$ bounds $Var(Z)$ as well. Using the definition of Z and linearity of expectation, it holds that:

$$\mathbb{E}[Z^2] = \mathbb{E}[(\sum_{i \neq j} \delta_{ij} Y_{ij})^2] \tag{45}$$
$$= \sum_{i \neq j, k \neq l} \delta_{ij} \delta_{kl} \mathbb{E}[Y_{ij} Y_{kl}] \tag{46}$$

We show now a bound for $\mathbb{E}[Y_{ij} Y_{kl}]$, for every $i \neq j$ and $k \neq l$. Consider any such pair Y_{ij}, Y_{kl}. Recall that $Y_{ij} = |\langle \vec{r}, \vec{\omega_{ij}} \rangle|$ and $Y_{kl} = |\langle \vec{r}, \vec{\omega_{kl}} \rangle|$.

In the general case, there is an angle ψ between $\vec{\omega_{ij}}$ and $\vec{\omega_{kl}}$. Note, that in the cases where $(i,j) = (k,l)$, or, if simply the direction vectors of two such difference vectors (between two different pairs of points) coincide, ψ equals 0. Again, since the uniform distribution on the d-dimensional sphere is invariant under orthogonal transformations, we may compute $\mathbb{E}[Y_{ij} Y_{kl}]$ assuming $\vec{w_{ij}} = \vec{e_d}$, the dth vector in the standard basis. As for $\vec{\omega_{kl}}$, we may assume it is of the form $\alpha \vec{e_d} + \beta \vec{e_{d-1}}$, where $\alpha = \sin \psi$, and $\beta = \cos \psi$.

As in the calculation of $f(d)$, we have:

$$\mathbb{E}[Y_{ij} Y_{kl}] = \int_{\mathbb{R}} \cdots \int_{\mathbb{R}} \frac{1}{(2\pi)^{d/2}} \left| \left\langle \frac{\vec{x}}{\|\vec{x}\|_2}, \vec{e_d} \right\rangle \right| \left| \left\langle \frac{\vec{x}}{\|\vec{x}\|_2}, \alpha \vec{e_d} + \beta \vec{e_{d-1}} \right\rangle \right| e^{\frac{-(\vec{x},\vec{x})}{2}} \partial \vec{x} \tag{47}$$

Under spherical coordinates the term $|\langle \frac{x}{\|\vec{x}\|_2}, \vec{\omega_{ij}} \rangle| |\langle \frac{x}{\|\vec{x}\|_2}, \vec{\omega_{kl}} \rangle|$ simplifies to $|\cos \theta_1| |\alpha \cos \theta_1 + \beta \sin \theta_1 \cos_{\theta_2}|$. Thus, evaluating the integral using spherical coordinates, we get:

$$\mathbb{E}[Y_{ij} Y_{kl}] = \int_{\mathbb{R}^+} \int_{[0,\pi)} \cdots \int_{[0,\pi)} \int_{[0,2\pi)} \frac{1}{(2\pi)^{d/2}} |\cos \theta_1| \, |\alpha \cos \theta_1 + \beta \sin \theta_1 \cos \theta_2| \, e^{\frac{-\rho^2}{2}} |J_T| \partial \theta_{d-1} \ldots \partial \theta_1 \partial \rho \tag{48}$$

Since $-1 \leq \alpha, \beta \leq 1$ and by using the triangle inequality, we have:

$$\mathbb{E}[Y_{ij}Y_{kl}] \leq \int\limits_{\mathbb{R}^+}\int\limits_{[0,\pi)}\dots\int\limits_{[0,\pi)}\int\limits_{[0,2\pi)} \frac{1}{(2\pi)^{d/2}}\left|\cos\theta_1\cos\theta_1\right| e^{\frac{-\rho^2}{2}}|J_T|\partial\theta_{d-1}\dots\partial\theta_1\partial\rho +$$
$$\int\limits_{\mathbb{R}^+}\int\limits_{[0,\pi)}\dots\int\limits_{[0,\pi)}\int\limits_{[0,2\pi)} \frac{1}{(2\pi)^{d/2}}\left|\cos\theta_1\sin\theta_1\cos\theta_2\right| e^{\frac{-\rho^2}{2}}|J_T|\partial\theta_{d-1}\dots\partial\theta_1\partial\rho \tag{49}$$

Substituting $\cos^2\theta_1 = 1-\sin^2\theta_1$, we now divide Eq. (49) by Eq. (42) as done in setction 3.6

$$\begin{aligned}
\mathbb{E}[Y_{ij}Y_{kl}] &\leq \frac{\int\limits_{[0,\frac{\pi}{2}]}(1-\sin^2\theta_1)\sin^{d-2}\theta_1\partial\theta_1}{A_{d-2}} + \frac{\int\limits_{[0,\frac{\pi}{2}]}\cos\theta_1\sin^{d-1}\theta_1\partial\theta_1\cdot\int\limits_{[0,\frac{\pi}{2}]}\cos\theta_2\sin^{d-2}\theta_2\partial\theta_2}{A_{d-2}\cdot A_{d-3}} \\
&= \frac{(1-\frac{d-1}{d})\cdot A_{d-2}}{A_{d-2}} + f(d)\cdot f(d-1) && (50)\\
&= \theta(\frac{1}{d}) && (51)
\end{aligned}$$

We use again the definition of A_k and the recursion in Eq. (44) for the first term, and as for the second term, we use an intermediate step in the calculation of $f(d)$ as in Eq. (43), as well as the bound $f(d) = \theta(\frac{1}{\sqrt{d}})$.

Returning to Eq. (45), we conclude the bound

$$Var(Z) = O\left(\frac{1}{d}\cdot\mathbb{S}(S)\right) \tag{52}$$

4 On Approximating the Diameter of a Set of Points

Given a set of n points in $\mathbb{R}^d$, we consider the problem of approximating the diameter of the set, which is defined to be the largest distance between any two points in the set. For the general case, the trivial algorithm works in time $O(dn^2)$. Of course, in the case $d = 1$, the problem can be solved in time $O(n)$ (by finding the leftmost and rightmost points).

After we came up with our randomized approximation algorithm to the problem, we found out that a derandomized version of it already exists (see Section 4 in [6]). Whereas the derandomized version of the algorithm involves an additional factor of about $(\pi d)^{d/2}$ queries, it guarantees the approximation factor. We survey our probabilistic algorithm that achieves a $(1+\varepsilon)$-approximation of the diameter with high probability and runs in time $O(d\varepsilon^{-(d-1)/2})$.

4.1 An Overview of the algorithm

we project the points onto a random direction and calculate the diameter of the projected 1-dimensional set of points. The pair of points that achieves the diameter of the projected points correspond to a pair of points in the original set, and the distance between the original points is at least the distance of the projected points. We repeat this process a sufficient number of times, to assure that at least one of the projections will be onto an axis that is close to the direction of the diameter vector.

For a specific pair of points that are a diameter, denote $\vec{v}$, as their difference vector. For any ε, there exists a small angle $\theta = \theta(\varepsilon)$ such that when projecting the points along any direction vector w such that the angle between v and w is less than θ, it holds that the length of the projection of the vector v is at least $(1-\varepsilon)\|\vec{v}\|$. Calculating the diameter of the projected points along that line (using the 1-dimensional algorithm) will yield a projected diameter of size at least $(1-\varepsilon)\|\vec{v}\|$. Since the length of a projected vector never exceeds the length of the vector itself, we get a desired approximation in this case.

The event that the angle between a random vector and a diameter vector is at most θ is not a typical event, and it occurs with probability $\Theta(d^{\frac{1}{2}}\varepsilon^{\frac{d-2}{2}})$. Hence, this experiment is repeated $O(\varepsilon^{-d/2})$ times.

References

[1] U. Feige. On sums of independent random variables with unbounded variance, and estimating the average degree in a graph. In *Proc. of the 36th STOC*, pages 594–603, 2004.

[2] O. Goldreich and D. Ron. Approximating Average Parameters of Graphs. 2005.

[3] G.E.P. Box, Mervin E. Muller. A Note on the Generation of Random Normal Deviates. In *Annals of Mathemafical Statistics*, Vol. 29, No. 2, pages 610–611, 1958.

[4] D.E. Knuth. *The Art of Computer Programming, Volume 2 Seminumerical Algorithms.* Addison-Wesley, 1981.

[5] C. Chen, F. Qi. The Best Bounds in Wallis' Inequality. In *Proc. of the American Mathematical Society*, Vol. 133, Number 2, Pages 397–401, 2004.

[6] P.K. Agarwal, J. Matousek, S. Suri. Farthest Neighbors, Maximum Spanning Trees and Related Problems in Higher Dimension. In *Proc. of WADS '91*, Pages 105–116, 1991.

Appendix A - The uniform distribution on the unit sphere in R^d

Our algorithm is described using independent continuous random variables chosen uniformly from $S^{d-1} \triangleq \{\vec{x} \in R^d : \|\vec{x}\| = 1\}$ (the d-dimensional unit sphere).

We now explain how to produce this distribution in a computational model in which we are given access to random variables chosen independently from the continuous uniform distribution on the unit interval.

We use a known method [4] in order to generate a d-dimensional unit vector using d independent identically distributed (denoted *I.I.D.*) normal random variables with expectation 0 and variance 1 (denoted $N(0,1)$). We then exploit the Box-Muller method [3] which transforms a pair of random variables *I.I.D.* uniformly on the unit interval (denoted $U[0,1]$) into a pair *I.I.D.* $N(0,1)$.

In order to choose a random unit vector from the unit sphere in $\mathbb{R}^d$ we may use a normally distributed vector in $\mathbb{R}^d$, and use its direction. Specifically:

Lemma 2. *Let $N_1, \ldots, N_d$ be I.I.D.d random variables distributed $N(0,1)$. The direction vector of the vector $(N_1, \ldots, N_d)$,*

$$(X_1, \ldots, X_d) \triangleq \frac{(N_1, \ldots, N_d)}{\sqrt{\sum_{j=1}^{d} {N_i}^2}}$$

is distributed uniformly on S^{d-1} .

Proof. It suffices to show that for every subset of S^{d-1}, the probabilty of the event that a vector was chosen from that set remains fixed under rotations (and in fact under any orthonormal transformation). We use the density function of the d-dimensional normal distribution,

$$f_{N_1,\ldots,N_d}(n_1, \ldots, n_d) = \frac{1}{(2\pi)^{d/2}} \cdot e^{\frac{-<\vec{n},\vec{n}>}{2}}$$

where $\vec{n} \triangleq (n_1, \ldots, n_d)^{\tau}$. Let $A \subseteq S^{d-1}$ be such a subset, and U the orthonormal matrix corresponding to such an orthonormal transformation. For a set $I \subseteq \mathbb{R}^d$, let us denote:

$$R(I) \triangleq \{\alpha\vec{v} : \vec{v} \in I, \alpha \in \mathbb{R}^+\} \quad ; \quad U(I) \triangleq \{U\vec{v} : \vec{v} \in I\}$$

We get:

$$
\begin{aligned}
\mathbb{P}r[\vec{X} \in U(A)] = & \quad \mathbb{P}r[\vec{N} \in R(U(A))] \\
= & \quad \mathbb{P}r[\vec{N} \in U(R(A)))] \\
= & \quad \int\limits_{U(R(A))} f_{n_1,\ldots,n_d}(\vec{n})\partial\vec{n} \\
= & \quad \int\limits_{U(R(A))} \frac{1}{(2\pi)^{d/2}} e^{\frac{-<\vec{n},\vec{n}>}{2}} \partial\vec{n} =^*
\end{aligned}
$$

We now change coordinates by the linear transformation:

$$(m_1, \ldots, m_d)^\tau = U^{-1}(n_1, \ldots, n_d)^\tau$$

Since U is orthonormal, it holds $|J_U| = \left|\frac{\partial\vec{n}}{\partial\vec{m}}\right| = 1$ and $< U\vec{m}, U\vec{m} >$ = $< \vec{m}, \vec{m} >$. It follows:

$$
\begin{aligned}
^* = & \quad \int\limits_{U^{-1}U(R(A))} \frac{1}{(2\pi)^{d/2}} e^{\frac{-<U\vec{m},U\vec{m}>}{2}} |J_U| \partial\vec{m} \\
= & \quad \int\limits_{R(A)} f_{n_1,\ldots,n_d}(\vec{m})\partial\vec{m} \\
= & \quad \mathbb{P}r[\vec{N} \in R(A)] \\
= & \quad \mathbb{P}r[\vec{X} \in A]
\end{aligned}
$$

□

Using the lemma, it is clear how to achieve such a distribution using d independent samples of the normal distribution. We now survey the question of simulating the normal distribution based on the uniform one.

In general, for any random variable X, it is possible to transform a uniform random variable $U[0,1]$into X, by applying $X^*(p)$ (X's quantile function) on $U[0,1]$, that is, $X^*(U[0,1])$ is distributed as X.

In the case of the normal distribution, the quantile function is not an elementary function, so that a different method should be considered. We achieve the normal distribution using the Box-Muller method, which transforms a pair of random variables *I.I.D.* $U[0,1]$into a pair *I.I.D.* $N(0,1)$. The method is based on a characteristic property of the 2-dimensional normal distribution, that in the polar representation of a 2-dimensional normally distributed vector, it holds that its angle Θ is distributed uniformly on $[0, 2\pi)$ and that the square of its radius R is distributed according to the exponential distribution with expectation $\frac{1}{2}$. The quantile function of the exponential distribution with expectation λ is $X^*(p) = -\frac{1}{\lambda}\log(p)$. This concludes to:

Lemma 3. *Let U_1, U_2 be two random variables I.I.D. $U[0,1]$. Then*

$$\begin{aligned} N_1 &\triangleq R\cos\Theta &= \sqrt{-2\log U_1}\cos(2\pi U_2) \\ N_2 &\triangleq R\sin\Theta &= \sqrt{-2\log U_1}\sin(2\pi U_2) \end{aligned}$$

are two random variables I.I.D. $N(0,1)$.

The composiotion of the two lemmas enables the generation of the uniform distribution on the unit sphere in a continous computational model given access a to source of uniformly distributed random variables on the unit interval.