Dimension Coupling: Optimal Active Learning of Halfspaces via Query Synthesis

Lin Chen¹, Hamed Hassani², and Amin Karbasi¹
¹ Department of Electrical Engineering, Yale University
² Computer Science Department, ETH Zurich

March 14, 2016

Abstract

In this paper, we consider the problem of actively learning a linear classifier through query synthesis where the learner can construct artificial queries in order to estimate the true decision boundaries. This problem has recently gained a lot of interest in automated science and adversarial reverse engineering for which only heuristic algorithms are known. In such applications, queries can be constructed de novo to elicit information (e.g., automated science) or to evade detection with minimal cost (e.g., adversarial reverse engineering).

We develop a general framework, called dimension coupling (DC), that 1) reduces a $d$-dimensional learning problem to $d-1$ low-dimensional sub-problems, 2) solves each sub-problem efficiently, and 3) appropriately aggregates the results and outputs a linear classifier. We consider the three most common scenarios in the literature: idealized noise-free, independent noise realizations, and agnostic settings. We show that the DC framework avoids the curse of dimensionality: its computational complexity in all three cases scales linearly with the dimension. Moreover, in the noiseless and noisy cases, we show that the query complexity of DC is near optimal (within a constant factor of the optimum algorithm). We also develop an agnostic variant of DC for which we provide strong theoretical guarantees. To further support our theoretical analysis, we compare the performance of DC with the existing work in all three settings. We observe that DC consistently outperforms the prior arts in terms of query complexity while often running orders of magnitude faster.

Introduction

In contrast to the passive model of supervised learning, where all the labels are provided without any interactions with the learning mechanism, the key insight in active learning is that the learning algorithm can perform significantly better if it is allowed to choose which data points to label. This approach has found far-reaching applications, including the classical problems in AI (e.g., classification [17], information retrieval [16], speech recognition [8]) as well as the modern ones (e.g., interactive recommender systems [10], optimal decision making [9]).
In statistical learning theory, one assumes that a set of hypotheses \( H \) along with a set of unlabeled data points \( X \) are given, where each data point \( x \in X \) is drawn i.i.d. from some distribution \( D \). Classical probably approximately correct (PAC) bounds then yield the sample complexity (i.e., the number of required i.i.d. examples) from \( D \) to output a hypothesis \( h \in H \) that will have estimation error at most \( \epsilon \) with probability at least \( 1 - \delta \), for some fixed \( \epsilon, \delta > 0 \). Here, the estimation error is defined as \( \text{error}(h) = \Pr_{x \sim D}[h(x) \neq h^*(x)] \), where \( h^* \) is the unknown true hypothesis. In the realizable case of learning a halfspace, i.e., when \( h^* \in \mathbb{R}^d \) perfectly separates the data points into positive and negative labels, it is known that with \( \tilde{O}(d/\epsilon) \) i.i.d. samples one can find a linear separator with an estimation error \( \epsilon \). In contrast, a simple counting argument based on sphere packing shows that any algorithm needs \( \Omega(d \log(1/\epsilon)) \) examples to achieve an estimation error of \( \epsilon \).

If one is not careful, active learning may require more samples than passive learning to achieve the same estimation error. One setting that is guaranteed to perform at least as well as passive learning is pool-based active learning \[12\]: a set of unlabeled examples are drawn i.i.d. where instead of obtaining all labels at once, the learning algorithm sequentially decides which labels to request and which ones to discard. The key challenge is to develop an algorithm that requests informative labels from the pool in such a way that the remaining labels can be inferred as quickly as possible. A principled way is to define a version space \( V \) containing all hypotheses consistent with the labels obtained so far and try to shrink it significantly by requesting new labels from the pool. An effective but computationally expensive method is the halving algorithm, also referred to as generalized binary search (GBS), where at each iteration the next example to label is chosen in such a way that it approximately cuts the version space by one half. The new version space is constructed and the process is repeated until the estimation error is less than \( \epsilon \). In many cases, computing the version space is computationally prohibitive. Dasgupta \[6\], in the idealized case that the labels are noise-free and there exists a realizable linear separator, showed that one requires \( \tilde{O}(d \log^2(1/\epsilon)) \) labels, an exponential improvement over \( O(d/\epsilon) \). Unfortunately, the algorithm achieving this bound is generally intractable. Perhaps the most common query methods that approximate the idealized halving algorithm are uncertainty sampling \[12\] and query-by-committee (QBC) \[7\]. Freund et al. \[7\] showed that QBC requests \( \tilde{O}(d \log(1/\epsilon)) \) labels, but comes at the price of prohibitive computational cost in each iteration. Similarly, Balcan et al. \[1\] showed that uncertainty sampling also achieves an exponential reduction in sample complexity, requiring \( \tilde{O}(d^2 \log(1/\epsilon)) \) labels. Unfortunately, most approximation methods are only guaranteed to work if the pool size grows exponentially fast in each iteration. Hence, one needs to always store a large amount of data in addition to computing a complex version space.

An attractive alternative to the pool-based framework is query synthesis \[3\]: a learner can request for any unlabeled data instance from the input space, including queries that the learner synthesizes from scratch. This way the pool size limitation is entirely eliminated. In many recent applications, ranging from automated science \[11\], to robotics \[5\], and to adversarial reverse engineering \[13\], query synthesis is the appropriate model. For instance, in security-sensitive applications (e.g., spam filters...
and intrusion detection systems) that routinely use machine learning tools, a growing concern is the ability of adversarial attacks to identify the blind spots of the learning algorithms. Concretely, classifiers are commonly deployed to detect miscreant activities. However, they are attacked by adversaries who generate exploratory queries to elicit information that in return allows them to evade detection [14]. In this work, we show how an adversary can use active learning methods by making synthetically de novo queries and thus identify the linear separator used for classification.

We should emphasize that in active learning via synthesized queries the learning algorithm can query the label of any points (irrespective of the underlying distribution $D$) in order to explore the hypothesis space. However, for evaluating the performance of the algorithm in terms of the estimation error, one may use the distribution $D$. This is quite different from pool-based framework where queries should be among the data points drawn i.i.d. from the underlying distribution $D$.

There is little known about the theoretical performance of actively learning a linear classifier via query synthesis. The closest work to our efforts is [2] where they propose a heuristic algorithm that approximates the version space by a convex body with promising empirical results in the noise-free setting. Their method is not appropriate for high dimensional data, as its computational complexity scales cubically with the dimension. Moreover, their algorithm fails in the presence of noise.

In this paper, we develop a framework, called Dimension Coupling (DC), with the following guarantees. In the noiseless setting, we show that its computational complexity is $\tilde{O}(d \log \frac{1}{\delta})$ and its query complexity is $\tilde{O}(d \log \frac{1}{\delta})$. Similarly, in the noisy setting, we show that the computational complexity of DC is $\tilde{O}(d(\log \frac{1}{\delta} + \log \frac{1}{\delta}))$ and its query complexity is $\tilde{O}(d(\log \frac{1}{\delta} + \log \frac{1}{\delta}))$. Note that in both settings the computational complexity scales linearly with the dimension. Moreover, the query complexity in both settings is near-optimal. Finally, we generalize our results to the agnostic case. Our empirical experiments demonstrate that DC runs orders of magnitude faster than the existing methods.

**Problem Formulation**

Our objective is to estimate an unknown halfspace $H^* = \{x \in \mathbb{R}^d : \langle h^*, x \rangle > 0 \}$, using as few queries as possible. Here, $\langle \cdot, \cdot \rangle$ is the standard inner product of the Euclidean space (also $\| \cdot \|$ denotes the Euclidean norm), $h^*$ is some (hidden) unit vector that we want to estimate, and a generic query is of the form $\text{sign}(\langle h^*, x \rangle)$ where $x$ is selected in $\mathbb{R}^d$. Let us emphasize here that the only information we obtain from a query is the sign of the inner product and not the value. E.g., the queries of the form $\text{sign}(\langle h^*, e_i \rangle)$, where $e_i$ is the $i$th standard basis vector, will only reveal the sign of the $i$th component of $h^*$ (and nothing further about its value).

In the noiseless setting, we observe the true outcome of the query, i.e. $\text{sign}(\langle h^*, x \rangle) \in \{1, -1\}$. In the noisy setting, the outcome is a flipped version of the true sign with independent flip probability $\rho$. That is, denoting the outcome by $Y$ we have $Y \in \{-1, 1\}$ and $\Pr[Y \neq \text{sign}(\langle h^*, x \rangle)] = \rho < 1/2$.

Since the length of the selected vector $x$ will not affect the outcome of the query, we only query the points on the unit sphere $S^{d-1} = \{x \in \mathbb{R}^d : \|x\| = 1\}$. Hence, we term
\( X = S^{d-1} \) as the query space. All possible unit vectors also reside on the unit sphere \( S^{d-1} \). Therefore, the initial version space, denoted by \( \mathcal{H} \), is also \( S^{d-1} \).

As aforementioned, to evaluate the performance of halfspace learning algorithms, we assume that unlabeled data points are drawn from the distribution \( D \) with a continuous probability density function \( f_D \). The estimation error is defined as \( \text{error}(h) = \Pr_{x \sim D}[h(x) \neq h^*(x)] = \int \mathbb{1}\{h(x) \neq h^*(x)\} f_D(x) dx \leq 4M_D \sinh(h - h^*]/2) \), where \( M_D = \max_x f_D(x) \). To make sure the estimation error error(\( h \)) \leq \epsilon \), it suffices to ensure \( |h - h^*| < \epsilon_D \), where \( \epsilon_D = 2 \sin (\epsilon/(4M_D)) \); e.g., when \( D \) is uniform, \( \epsilon_D = 2 \sin (\pi \epsilon/2) \). Therefore, henceinafter we only discuss how to guarantee \( |h - h^*| < \epsilon_D \) with high probability (say, at least \( 1 - \delta \)). We report all the results in terms of \( |h - h^*| \) but it is easy to convert them to the estimation error. Thus, given \( \epsilon, \delta > 0 \), we seek an algorithm that (i) adaptively selects vectors \( x_1, x_2, \ldots \), (ii) observes the (noisy) responses to each query sign \( h^*, x_i \), (iii) and outputs, using as few queries as possible, an estimate \( \hat{h} \) of \( h^* \) such that \( |\hat{h} - h^*| < \epsilon \) with probability at least \( 1 - \delta \).

### Dimension Coupling Based Framework

Suppose that \( h^* \) has the form \( h^* = \sum_{i=1}^d c_i e_i \), where \( \{e_i\}_{i=1}^d \) is an arbitrarily chosen orthonormal basis for \( \mathbb{R}^d \). We assume w.l.o.g. that \( h^* \) is normalised (i.e., \( \sum_{i=1}^d c_i^2 = 1 \)).

Our objective is then to learn the coefficients \( \{c_i\}_{i=1}^d \) within a given precision by using the (noisy) responses to the selected sign queries. The key insight here is that this task can be partitioned in a divide-and-conquer fashion into many smaller tasks, each involving a few dimensions. The final answer (the values of \( \{c_i\}_{i=1}^d \)) will then be obtained by aggregating the answers of these subproblems.

**Example 1.** Assume \( h^* = c_1 e_1 + c_2 e_2 + c_3 e_3 + c_4 e_4 \), where \( e_i \)'s are the standard basis vectors for \( \mathbb{R}^4 \). Define

\[
\hat{e}_1 = \frac{c_1 e_1 + c_2 e_2}{\sqrt{c_1^2 + c_2^2}}, \quad \hat{e}_2 = \frac{c_3 e_3 + c_4 e_4}{\sqrt{c_3^2 + c_4^2}}.
\]

Note here that \( \hat{e}_1 \) is the (normalised) orthogonal projection of \( h^* \) onto \( \text{span}\{e_1, e_2\} \) and \( \hat{e}_2 \) is the (normalised) orthogonal projection of \( h^* \) onto \( \text{span}\{e_3, e_4\} \). Consider the following procedure to learn \( h^* \): first find out what \( \hat{e}_1 \) and \( \hat{e}_2 \) are, and then use the relation \( h^* = \sqrt{c_1^2 + c_2^2} \hat{e}_1 + \sqrt{c_3^2 + c_4^2} \hat{e}_2 \) to find \( h^* \) based on the orthonormal vectors \( \hat{e}_1, \hat{e}_2 \). By this procedure, the original “four-dimensional” problem has been broken into three “two-dimensional” problems.

For general \( d \), the idea is similar: We break the problem into at most \( d - 1 \) “two-dimensional” problems that each can be solved efficiently. More formally, let us assume that we have an algorithm, called DC\( ^2_c \) \( (e_1, e_2, \epsilon, \delta) \), that takes as input two orthonormal vectors \( e_1, e_2 \) and outputs with probability at least \( 1 - \delta \) a vector \( \hat{e} \) with the following three properties:

\[
\hat{e} \in \text{span}\{e_1, e_2\}, \quad \|\hat{e}\| = 1, \quad \|\hat{e} - \frac{(h^*, e_1)(h^*, e_2)}{(h^*, e_1)(h^*, e_2)} e_1\| < \epsilon.
\]

In other words, the unit vector \( \hat{e} \) is within a distance \( \epsilon \) to the (normalised) projection of \( h^* \) onto the subspace \( \text{span}\{e_1, e_2\} \). In the next section, we will explain in detail how
to design an optimal candidate for DC^2 that uses (noisy) responses to queries of the form \( \langle x, h^* \rangle \). In the current section, we explain a framework DC that estimates \( h^* \) using at most \( d-1 \) calls to DC^2 (a formal description is given in Algorithm 1). Consider the decomposition \( h^* = \sum_{i=1}^{d} c_i e_i \). For simplicity assume that \( d \) is an even number. We can write

\[
\hat{h}^* = \sum_{i=1}^{d} c_i e_i = \sum_{j=1}^{d/2} \hat{c}_j \frac{c_{2j-1} e_{2j-1} + c_{2j} e_{2j}}{\sqrt{c_{2j-1}^2 + c_{2j}^2}},
\]

where in the last step we have taken \( \hat{c}_j \triangleq \sqrt{c_{2j-1}^2 + c_{2j}^2} \). Now, note that \( \sum_{j=1}^{d/2} \hat{c}_j \hat{e}_j \) is the (normalised) orthogonal projection of \( h^* \) onto span\( \{ e_{2j-1}, e_{2j} \} \). Hence, by using DC^2 \( (e_{2j-1}, e_{2j}, \epsilon, \delta) \) we can obtain, with probability at least \( 1 - \delta \), a good approximation \( \hat{e}_j \) (within a distance \( \epsilon \)) of this projection. Therefore, for small enough \( \epsilon \) we have \( h^* \approx \sum_{j=1}^{d/2} \hat{c}_j \hat{e}_j \).

The idea is then to repeat the same procedure as in (1) to the newly obtained representation of \( h^* \). Hence, by repeating this procedure at most \( \log_2 d \) times we will reach a vector which is the final approximation of \( h^* \).

**Algorithm 1** Dimension Coupling (DC)

Input: an orthonormal basis \( E = \{ e_1, e_2, \ldots, e_d \} \) of \( \mathbb{R}^d \).

while \( d > 2 \)

1. for \( j = 1 \) to \( \lfloor d/2 \rfloor \)
   
   Replace the two vectors \( e_{2j-1}, e_{2j} \) in \( E \) with the vector DC^2 \( (e_{2j-1}, e_{2j}, \epsilon, \delta) \).

end for

2. Set \( d \leftarrow \lfloor d/2 \rfloor \).

end while

return DC^2 \( (e_1, e_2, \epsilon, \delta) \).

**Theorem 2.** For DC (outlined in Algorithm 1) we have:

1. DC will call the two-dimensional subroutine DC^2 at most \( d-1 \) times.

2. Provided that the output of DC^2 is with probability \( 1 - \delta \) within distance \( \epsilon \) of the true value and \( \epsilon \leq \frac{1}{6d} \), DC ensures an estimation error of at most \( 6d \epsilon \) with probability at least \( 1 - 6d \delta \).

We defer the proof of all theorems to the appendix. As a result of Theorem 2 if we desire the framework DC to estimate \( h^* \) within distance \( \epsilon \) and with probability at least \( 1 - \delta \), then it is enough to fix the corresponding parameters of DC^2 to \( \epsilon = \frac{\epsilon}{6d} \) and \( \delta = \frac{\delta}{d} \).
DC$^2$: Solving in 2 Dimensions

Before illustrating the algorithm DC$^2$, let us review some notation. Given two orthonormal vectors $e_1, e_2$ we denote the (normalised) projection of $h^*$ onto $\text{span}\{e_1, e_2\}$ by $h^\perp$, i.e.,

$$h^\perp = \frac{(h^*, e_1) e_1 + (h^*, e_2) e_2}{\| (h^*, e_1) e_1 + (h^*, e_2) e_2 \|_2}. \quad (2)$$

The objective of DC$^2$ ($e_1, e_2, \epsilon, \delta$) is to find a unit vector $\hat{e} \in \text{span}\{e_1, e_2\}$ such that $\| \hat{e} - h^\perp \| < \epsilon$. In fact, we require the latter to hold with probability at least $1 - \delta$.

Any unit vector inside $\text{span}\{e_1, e_2\}$, e.g., $h^\perp$, can equivalently be represented as a pair $(c_1, e_2)$ on the two-dimensional unit circle $S^1$ (e.g., $h^\perp = c_1 e_1 + c_2 e_2$ and $c_1^2 + c_2^2 = 1$). To simplify notation, we use a point $(c_1, e_2) \in S^1$ and its corresponding unit vector $c_1 e_1 + c_2 e_2$ interchangeably. In this setting, it is easy to see that for any $x \in \text{span}\{e_1, e_2\}$

$$\text{sign} \langle x, h^\perp \rangle = \text{sign} \langle x, h^\perp \rangle. \quad (3)$$

We take a Bayesian approach. In the beginning, when no queries have been performed, DC$^2$ assumes no prior information about the vector $h^\perp$. Therefore, it takes the uniform distribution on $S^1$ (with pdf $p_{0}(h) = \frac{1}{2\pi}$) as its prior belief about $h^\perp$. After performing each query, the posterior (belief) about $h^\perp$ will be updated according to the observation. We let $p_m(h)$ denote the (pdf of the) posterior after performing the first $m$ queries. In this manner, DC$^2$ runs in total of $T_{\epsilon, \delta}$ rounds, where in each round a specific query is selected and posed to the oracle. The number $T_{\epsilon, \delta}$ will be specified later (see Theorems 3 and 4). Upon the completion of round $T_{\epsilon, \delta}$, the algorithm returns as its final output a vector $\hat{e} \in S^1$ that maximises the posterior pdf $p_{T_{\epsilon, \delta}}(h)$. If there are multiple such maximisers, it picks one arbitrarily. We now proceed with a detailed description of DC$^2$ (a formal description is provided in Algorithm 2). We first consider the simpler noise-free case and the other settings will follow afterwards.

**Noise-Free Case:** We explain DC$^2$ (outlined in Algorithm 2) with the help of a running example given in Figure 1. As we will see, after each round of DC$^2$ the possible region that $h^\perp$ can belong to will be “halved”.

We first note that as the initial distribution $p_0$ is assumed to be the uniform distribution on $S^1$, the vector $x_1$ (see step 2-(a) of Algorithm 2) can indeed be any point on the unit circle $S^1$. Thus, DC$^2$ chooses $x_1$ arbitrarily on $S^1$. By (3), using the query sign $\langle x_1, h^\perp \rangle$ will also give us the value of sign $\langle x_1, h^\perp \rangle$. Depending on this value, it is easy to verify that only half of $S^1$ can possibly contain $h^\perp$ (see Figure 1). Let us denote this region by $R_1$. Hence, the probability distribution $p_1(h)$ (which is our current belief about $h^\perp$) is updated as follows: for $h \in R_1$ we have that $p_1(h) = 0$, and as all the points inside the half-circle $R_1$ are equiprobable, we have for $h \in R_1$ that $p_1(h) = 1/\pi$. In other words, at time $m = 1$ the vector $h^\perp$ could have been anywhere on the unit circle, but, after round $m = 1$ it can only belong to the half-circle $R_1$. Thus, after the first round, DC$^2$ “halves” the admissible region of $h^\perp$. Continuing in this theme, it is not hard to verify that (see Figure 1) at round $m = 2$ the value of $p_2(h)$ is non-zero and uniform only on a region $R_2$ which is a quarter-circle. In an inductive manner, letting $R_{m-1}$ denote the admissible region (sector) at round $m - 1$ (see Figure 1) and assuming that $p_{m-1}$ is only non-zero and uniform on the sector $R_{m-1}$, then $x_m$ at round $m$ is
Algorithm 2 DC$^2$

Input: orthonormal vectors $e_1, e_2$, estimation error at most $\epsilon$, success probability at least $1 - \delta$.

1. Set $p_0(h)$ to be uniform, i.e., $\forall h \in S^1 : p_0(h) = 1/2\pi$.

2. for $m = 1$ to $T_{\epsilon, \delta}$
   
   (a) find a vector $x_m \in S^1$ which is a solution to the following equation:
   $$\int_{S^1} \text{sign} \langle x, h \rangle p_{m-1}(h) dh = 0.$$ If there are multiple solutions, choose one arbitrarily.

   (b) ask from the oracle the value of $\text{sign} \langle x_m, h^* \rangle$.

   (c) based on the response obtained from the oracle, update the distribution $p_{m-1}(h)$ to $p_m(h)$.

end for

3. return $\hat{e} = \arg \max_{h \in S^1} p_{T_{\epsilon, \delta}}(h)$.

---

Figure 1: An example to illustrate DC$^2$ in the noiseless setting. In the first round, $x_1$ is arbitrarily chosen on $S^1$. For the choice in the figure, we have $\text{sign} \langle x_1, h^* \rangle = \text{sign} \langle x_1, h^\perp \rangle = -1$. For any point $h$ above the red line we have that $\text{sign} \langle x, h \rangle = -1$ and for the points outside this half-circle the result is +1. Therefore, the distribution (pdf of) $p_1$ is uniform on the region above the red line and is zero below it. For round $m = 2$ it is easy to see that the direction of $x_2$ should be along the red line. For $x_2$ chosen as in the figure, we have $\text{sign} \langle x_2, h^* \rangle = +1$ and hence at the end of the second round DC$^2$ concludes that the vector $h^\perp$ could uniformly be any point inside $R_2$. In a generic round $m$, any vector orthogonal to the mid-point of sector $R_{m-1}$ can be considered as a candidate for $x_m$. For the choice in the figure, we have $\text{sign} \langle x_m, h^\perp \rangle = -1$. Thus, at the end of round $m$, DC$^2$ concludes that $h^\perp$ can uniformly be any point inside $R_m$.

precisely the vector that is orthogonal to the midpoint of the sector $R_{m-1}$. Therefore, after observing the value of $\text{sign} \langle x_m, h^* \rangle$, the admissible region $R_m$ is the better half of $R_{m-1}$ that is compatible with the observation (i.e., it contains $h^\perp$). Also, $R_m$ is again a sector and $p_m$ will be uniform on $R_m$ and zero outside. It is also easy to see that the circular angle for the sector $R_m$ is $\frac{\pi}{2m}$. The following statement is now immediate.

**Theorem 3.** Consider DC in the absence of noise ($\rho = 0$). If we let $T_{\epsilon, \delta} = \lceil \log_2 \frac{1}{\epsilon} \rceil$, then it outputs a vector that is within a distance $\epsilon$ of $h^\perp$. 7
A few comments are in order: The above guarantee for DC$^2$ holds with probability one and thus the parameter $\delta$ is irrelevant in the noiseless setting. Furthermore, during each round of DC$^2$, the distribution $p_m$ can be represented by only two numbers (the starting and ending points of the sector $R_m$), and the vector $x_m$ can be computed efficiently (it is the orthogonal vector to the midpoint of $R_m$). Therefore, assuming one unit of complexity for performing the queries, DC$^2$ can be implemented with complexity $O(T_{r,\delta})$. Finally, by using Theorem 2 we conclude that DC requires $O(d \log \frac{1}{\epsilon})$ queries with computational complexity $O(d \log \frac{1}{\delta})$.

**Noisy Case:** In general, DC$^2$ follows a similar procedure as in the noiseless case except that the distributions $p_m$ does not look as simple. However, as we now discuss, these distributions can still be stored efficiently and as a result the vector $x_m$ can be computed efficiently. Indeed, (the pdf of) $p_m$ is piecewise constant on the unit circle (see Figure 2). More precisely, at any round $m$, there are at most $2m$ points $u_1, u_2, \ldots, u_{2m}$ that are ordered clockwise on the unit circle and $p_m$ is constant when restricted to each of the sectors $[u_i, u_{i+1})$. At round $m + 1$, in order to find $x_{m+1}$ (see step 2-a) of Algorithm 2, DC$^2$ first finds a line that passes through the centre of $S^1$ and cuts $S^1$ into two “halves” which have the same measure with respect to $p_m$. Note that finding such a line can be done in $O(m)$ steps because $p_m$ has the piecewise constant property. Once such a line is found, it is then easy to see that $x_{m+1}$ can be any of the two points orthogonal to the line. As a result, DC$^2$ at round $m + 1$ can find $x_{m+1}$ in $O(m)$ operations. We denote the half-circle containing $x_{m+1}$ by $R^+$ and the other half by $R^-$. We refer to Figure 2 for a schematic illustration. Once a noisy response to the query $\langle x_{m+1}, h^* \rangle$ is obtained, the probability distribution $p_m$ will be updated to $p_{m+1}$ in the following way. First, consider the event that the outcome of $\langle x_{m+1}, h^* \rangle$ is $+1$. We have $p_m(\langle x_{m+1}, h^* \rangle = +1) = (1 - \rho) p_m(R^+) + \rho p_m(R^-) = 1/2$, and similarly $p_m(\langle x_{m+1}, h^* \rangle = +1) = 1/2$. Therefore, by Bayes theorem we obtain the following update rules for $p_{m+1}$. If we observe that $\langle x_{m+1}, h^* \rangle = +1$, then for $h \in R^+$ we have $p_{m+1}(h) = (2(1 - \rho)p_m(h)$ and for $h \in R^-$ we have $p_{m+1}(h) = (2\rho)p_m(h)$. Also, if we observe that $\langle x_{m+1}, h^* \rangle = -1$, then for $h \in R^+$: $p_{m+1}(h) = (2\rho)p_m(h)$ and for $h \in R^-: p_{m+1}(h) = (2(1 - \rho)p_m(h)$. (note that the factor of 2 here is due to the normalization.) It is easy to verify that $p_{m+1}$ is also a piecewise constant distribution (now on $2(m + 1)$ sectors; see Fig. 2).

**Theorem 4.** In the noisy setting (with independent flip probability $\rho$), having

$$T_{r,\delta} \geq M + \max\{T_0, T_1, T_2, T_3\} = O(\log \frac{1}{\epsilon} + \log \frac{M}{\delta})$$

is sufficient to guarantee that DC$^2$ outputs with probability at least $1 - \delta$ a vector that is within a distance $\epsilon$ of $h^1$. Here, we have $M = \lceil \frac{2\log \frac{4}{\log(4\rho(1-\rho))}}{\log(4\rho(1-\rho))} \rceil$, $T_0 = \frac{8\log \frac{4}{\log(4\rho(1-\rho))}}{\log(2(1-\rho))}$, $T_1 = \frac{8\log \frac{4}{\log(2(1-\rho))}}{\log(2(1-\rho))}$, $T_2 = \frac{8\log \frac{4}{\log(2(1-\rho))}}{\log(2(1-\rho))}$, $T_3 = \frac{24\rho \log \frac{4}{\log(2(1-\rho))}}{\log^2(2(1-\rho))} (\log(M) + \log(\frac{1}{\delta})).$

Theorem 2 indicates that DC requires $\tilde{O}(d \log \frac{1}{\epsilon} + \log \frac{1}{\delta})$ queries. Also, as discussed above, the computational complexity of DC$^2$ is $O(T_{r,\delta})$. Hence, DC has computational complexity $\tilde{O}(d \log \frac{1}{\epsilon} + \log \frac{1}{\delta})^2).$
Upon the completion of round $m$ (left figure), the distribution (pdf of) $p_m$ is constant over each of the sectors $[u_i, u_{i+1})$. In the next round (right figure), in order to find $x_{m+1}$, DC first finds a diagonal line (red line) which separates two half-circles ($R^+$ and $R^-$) that each has measure $1/2$ w.r.t $p_m$. The vector $x_{m+1}$ will then be one of the two points on the unit circle that are orthogonal to this line. For updating $p_m$ to $p_{m+1}$, we note that all the points inside $R^+$ get the same factor (either $2\rho$ or $2(1-\rho)$ depending on the outcome of the query). The same is true for $R^-$. Thus, $p_{m+1}$ is again a piecewise constant pdf but now on $2(m+1)$ sectors.

**Agnostic Case:** A common approach used in the agnostic setting is the empirical risk minimization (ERM), which generates queries by independently sampling from the marginal distribution of $X$, i.e., $P_X$, and then selects a unit vector that minimizes the number of errors made on these $n$ queries. We follow a similar approach as in [15]. Suppose that we are given a query budget of $n$. We allocate $n/3$ queries to DC and let $h_1$ denote the unit vector selected by DC. Then we allocate $n/3$ queries to ERM and let $h_2$ denote the unit vector proposed by ERM using its $n/3$ queries. Consider the region $\Delta$ that $h_1$ and $h_2$ disagree on, i.e., $\Delta = \{x \in S^{d-1} : h_1(x) \neq h_2(x)\}$. Let $P_\Delta$ be the restriction of the probability measure $P_X$ on $\Delta$. Here we sample the remaining $n/3$ queries from $\Delta$ according to $P_\Delta$. We write $\hat{R}_\Delta(h_i)$ for the average number of errors made by $h_i$ on the sampled $n/3$ queries. Finally, we set $R_\Delta(h_i) = E[\hat{R}_\Delta(h_i)]$. The algorithm will output $\hat{h} = \text{arg min}_{h \in \{h_1, h_2\}} \hat{R}_\Delta(h)$ as the final result.

**Theorem 5.** Using the above procedure we have the following bounded error probability in expectation $E[R(\hat{h})] \leq \min\{E[R(h_1)], E[R(h_2)]\} + 2\sqrt{\frac{1}{n \epsilon}},$ where $n$ is the number of queries and $\epsilon$ is the base of the natural logarithm.

**Empirical Results**

In this section, we extensively evaluate the performance of DC against the following baselines:

- **Random-Sampling:** Queries are generated by sampling uniformly at random from the unit sphere $S^{d-1}$.

- **Uncertainty-Sampling:** Queries are sampled uniformly at random from the orthogonal complement of $w$, where $w$ is the vector learned by linear SVM.
Figure 3: Figures 3a, 3b, and 3c show the estimation error as we increase the number of queries, for \( d = 25, 50, 100 \), respectively. Fig. 3d shows the corresponding execution times. Fig. 3e and 3f show the estimation error as we increase the number of queries for \( d = 25, 50 \) and the noise level \( \rho = 0.1 \). The corresponding execution times are shown in Fig. 3g. Fig. 3h presents the estimation error of DC and Repetitive-DC as we increase the number of queries for \( d = 1000 \) and noise levels \( \rho = 0.01, 0.1, 0.2 \). Finally, Fig. 3i demonstrates the error probability w.r.t. the number of queries in the agnostic setting for \( d = 25 \).

- **QUERY-BY-BAGGING**: The bag size is set to 20 and 1000 queries are generated at each iteration. The query with the largest disagreement is picked \([1]\).

- **SPECTRAL**: The version space is approximated by the largest ellipsoid consistent with all previous query-label pairs. Then, at each iteration a query is selected to approximately halve the ellipsoid \([2]\).

- **REPEITIVE-DC**: In the noisy setting, one easy way to apply DC is to query each point \( R \) times and use the majority rule to determine its label.

Our metrics to compare different algorithms are: a) estimation error, b) query complexity, and c) execution time. In particular, as we increase the number of queries we
measure the average estimation errors and execution times for all the baselines (with 90% confidence intervals). By nature, in active learning via query synthesis, all data points and queries are generated synthetically. For all the baselines, we used the fastest available implementations in MATLAB.

**Noise-free setting:** Figures 3a, 3b and 3c (with dimension $d = 25, 50, 100$, respectively) show that in terms of estimation error, DC outperforms all other baselines, and significantly outperforms RANDOM-SAMPLING, UNCERTAINTY-SAMPLING and QUERY-BY-BAGGING. Note that the estimation errors are plotted in log-scales. In terms of execution times, we see in Fig. 3d that DC runs three orders of magnitude faster than other baselines. Training an SVM at each iteration for RANDOM-SAMPLING, UNCERTAINTY-SAMPLING and QUERY-BY-BAGGING comes with a huge computational cost. Similarly, SPECTRAL requires solving a convex optimization problem at each iteration; thus its performance drastically deteriorates as the dimension increases, which makes it infeasible for many practical problems.

**Noisy setting:** We set the noise level to $\rho = 0.1$ and compare the performance of DC against RANDOM-SAMPLING, UNCERTAINTY-SAMPLING, QUERY-BY-BAGGING, and REPETITIVE-DC (for $R = 5$). As mentioned in [2], and we have also observed in our experiments, SPECTRAL does not work even for small amounts of noise as it incorrectly shrinks the version space and misses the true linear separator. We see again in Figures 3e and 3f (for $d = 25, 50$) that DC significantly outperforms all other methods in terms of estimation error as we increase the number of queries. Figure 3g shows that DC still runs $\sim 100$ times faster than RANDOM-SAMPLING, UNCERTAINTY-SAMPLING, and QUERY-BY-BAGGING. Clearly, DC has a higher computational cost than REPETITIVE-DC, as DC performs a Bayesian update after each query. Finally, as we increase the dimension to $d = 1000$, RANDOM-SAMPLING, UNCERTAINTY-SAMPLING, and QUERY-BY-BAGGING become significantly slower. Hence, in Fig. 3h we only show how the estimation error (for noise levels $\rho = 0.01, 0.1, 0.2$) decreases for DC and REPETITIVE-DC with more queries.

**Agnostic setting:** We sample a set of size 1000 from the Von Mises–Fisher distribution (the analog of the normal distribution on $S^{d-1}$) and compute the empirical error. The noise level at each point is drawn independently from the truncated normal $\mathcal{N}(0.1, 0.5)$ on $[0, 0.5]$. Fig. 3i shows the error probability of RANDOM-SAMPLING, UNCERTAINTY-SAMPLING, QUERY-BY-BAGGING and AGNOSTIC-DC (for $d = 25$). Both AGNOSTIC-DC and UNCERTAINTY-SAMPLING achieve an error probability $\sim 0.1$ after 700 queries. However, as shown by Theorem 5 AGNOSTIC-DC has a strong theoretical guarantee which, by contrast, other baselines do not have.
References

[1] N. Abe and H. Mamitsuka. Query learning strategies using boosting and bagging. In ICML, page 1. Morgan Kaufmann Pub, 1998.

[2] I. Alabdulmohsin, X. Gao, and X. Zhang. Efficient active learning of halfspaces via query synthesis. In AAAI 2015, 2015.

[3] D. Angluin. Queries and concept learning. Machine learning, 1988.

[4] M.-F. Balcan, A. Broder, and T. Zhang. Margin based active learning. In Learning Theory, pages 35–50. Springer, 2007.

[5] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. JAIR, 1996.

[6] S. Dasgupta. Analysis of a greedy active learning strategy. Advances in neural information processing systems, 17:337–344, 2005.

[7] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. Machine learning, 1997.

[8] D. Hakkani-Tur, G. Riccardi, and A. Gorin. Active learning for automatic speech recognition. In ICASSP), volume 4, pages IV–3904. IEEE, 2002.

[9] S. Javdani, Y. Chen, A. Karbasi, A. Krause, J. A. Bagnell, and S. Srinivasa. Near optimal bayesian active learning for decision making. AISTAT, 2014.

[10] A. Karbasi, S. Ioannidis, and L. Massoulie. Comparison-based learning with rank nets. ICML, 2012.

[11] R. D. e. a. King. The automation of science. Science, 2009.

[12] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, 1994.

[13] D. Lowd and C. Meek. Adversarial learning. In KDD, pages 641–647. ACM, 2005.

[14] B. Nelson, B. I. Rubinstein, L. Huang, A. D. Joseph, S. J. Lee, S. Rao, and J. Tygar. Query strategies for evading convex-inducing classifiers. JMLR, 2012.

[15] R. Nowak. Noisy generalized binary search. Advances in neural information processing systems, 21:1366–1374, 2009.

[16] S. Tong and E. Chang. Support vector machine active learning for image retrieval. In Proceedings of the 9th ACM international conference on Multimedia, pages 107–118. ACM, 2001.

[17] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. JMLR, 2:45–66, 2002.
Appendix A: Proof of Theorem 2

At each round of DC, the value of $d$ is replaced by $\lceil d/4 \rceil$. It is thus not hard to verify that DC runs in $\lceil \log_2 d \rceil$ rounds until $d \leq 2$ and in total there are at most $d - 1$ usages of $DC^2$. As a result, if the probability of success for $DC^2$ is at least $1 - \delta$, then by the union bound the probability of success of DC is at least $1 - (d - 1)\delta$.

For the last part of the theorem, we prove a more general statement: Assume that we run DC with an input being an orthonormal set $\{e_1, e_2, \ldots, e_d\}$ where $e_i, h^* \in \mathbb{R}^K$ for some $K \geq d$. We prove that DC outputs a vector that is close to the (normalised) orthogonal projection of $h^*$ into $\text{span}\{e_1, e_2, \ldots, e_d\}$. More precisely, we define

$$h^* = \frac{\sum_{i=1}^{d} (e_i, h^*) e_i}{\|e_i, h^* e_i\|}.$$ (5)

Then, DC runs in $\log_2 d$ rounds, calls $DC^2$ $d - 1$ times, and outputs with probability at least $1 - \delta d$ a vector $\hat{h}$ for which $\|h^* - \hat{h}\| < 6\delta d$. In exactly similar way as discussed above, we can conclude that DC runs in $\log_2 d$ times and uses $DC^2$ $d - 1$ times. Also, again by the union bound, with probability at least $1 - \delta d$ all the outputs of $DC^2$ are a close estimate (within distance $\epsilon$) of their corresponding objective. Thus, by assuming that all the calls of $DC^2$ have been successful (which happens w.p. at least $1 - \delta d$), we use an inductive argument to prove that $\|h^* - \hat{h}\| < 6\delta d$. We use induction on $d$. For $d = 2$ the result is clear. We now prove the result when $d = k$ assuming that it holds for all $d < k$. For simplicity, we assume that $k$ is an even number, i.e., $k = 2t$ (the proof follows very similarly for $k$ being odd). We can then write

$$h^* = \sum_{i=1}^{d} c_i e_i = \sum_{j=1}^{t} \hat{c}_j h_j^1,$$ (6)

where $\hat{c}_j = \sqrt{c_{2j-1}^2 + c_{2j}^2}$ and $h_j^1 = \frac{c_{2j-1} e_{2j-1} + c_{2j} e_{2j}}{\sqrt{c_{2j-1}^2 + c_{2j}^2}}$. Note that $h_j^1 = \frac{c_{2j-1} e_{2j-1} + c_{2j} e_{2j}}{\sqrt{c_{2j-1}^2 + c_{2j}^2}}$ is precisely the (normalised) orthogonal projection of $h^*$ (and also $h^1$) onto $\text{span}\{e_{2j-1}, e_{2j}\}$. As we explained in Section 2, in the first round of DC each vector $h_j^1$ will be replaced by the output of $DC^2$ ($e_{2j-1}, e_{2j}, \epsilon, \delta$) which we denote by $\hat{e}_j$. Let us now define the vector $\hat{h}^1$ as

$$\hat{h}^1 = \frac{\sum_{j=1}^{t} (h_j^1, \hat{e}_j) \hat{e}_j}{\sqrt{\sum_{j=1}^{t} (h_j^1, \hat{e}_j)^2}}.$$ 

It is easy to verify that $\|\hat{h}^1\| = 1$ as $\{\hat{e}_1, \hat{e}_2, \ldots, \hat{e}_t\}$ is an orthonormal set. By the assumption of the induction, the final output of DC, which we denote by $\hat{h}$, will be within the distance $6\epsilon t$ of $\hat{h}^1$. That is,

$$\|\hat{h}^1 - \hat{h}\| < 6\epsilon t.$$ (7)

We now prove that

$$\|h^1 - \hat{h}\| < 6\epsilon t.$$ (8)
From (7) and (8) the induction hypothesis will be immediate as we can write

\[ ||h^k - \hat{h}^k|| \leq ||h^k - h^1|| + ||h^1 - \hat{h}^1|| < 6\epsilon t + 6\epsilon t = 6\epsilon. \]

It thus remains to prove (8).

Firstly, we define \( \beta \triangleq \sqrt{\sum_{j=1}^{t} (h^j, \hat{e}_j)^2} \). We have

\[
||h^k - h^1|| = \| h^k - \frac{\sum_{j=1}^{t} (h^j, \hat{e}_j) \hat{e}_j}{\beta} \|
= \| \beta h^k - \frac{\sum_{j=1}^{t} (h^j, \hat{e}_j) \hat{e}_j}{\beta} \|
= \| (\beta - 1) h^k + h^1 - \frac{\sum_{j=1}^{t} (h^j, \hat{e}_j) \hat{e}_j}{\beta} \|
\leq \| 1 - \frac{\beta}{\beta} \| + \| \frac{h^1 - \sum_{j=1}^{t} (h^j, \hat{e}_j) \hat{e}_j}{\beta} \|. \tag{9}
\]

Secondly, we have

\[
|\beta^2 - 1| = \| \sum_{j=1}^{t} (h^j, \hat{e}_j)^2 - (h^k, h^1)^2 \|
\leq \sum_{j=1}^{t} \| (h^j, \hat{e}_j) - (h^k, h^1)^2 \| (h^k, \hat{e}_j) + (h^k, h^1)^2 \|
\leq \sum_{j=1}^{t} \| h^j \| \cdot \| \hat{e}_j - h^1 \| \cdot (\| h^k \| \cdot \| \hat{e}_j \| + \| h^k \| \cdot \| h^1 \|)
\leq 2\epsilon t,
\]

where the last step follows from \( \| h^k \| = \| h^1 \| = \| \hat{e}_j \| = 1 \) and \( \| \hat{e}_j - h^1 \| \leq \epsilon \). Hence, by noting the fact that \( 2\epsilon t = \epsilon d \leq \frac{1}{6} \) we obtain

\[ \beta \in [\sqrt{1 - 2\epsilon t}, \sqrt{1 + 2\epsilon t}], \tag{10} \]

and

\[ \left| \frac{1 - \beta}{\beta} \right| \leq \max\left\{ \frac{1}{\sqrt{1 - 2\epsilon t}} - 1, 1 - \frac{1}{\sqrt{1 + 2\epsilon t}} \right\} \leq 2\epsilon t. \tag{11} \]

Fourthly, similar as above we can show that

\[ ||h^k - \sum_{j=1}^{t} (h^j, \hat{e}_j) \hat{e}_j|| = \| \sum_{j=1}^{t} (h^j, h^j) h^1_j - (h^k, \hat{e}_j) \hat{e}_j || \leq 2\epsilon t. \tag{12} \]

Now, by plugging (10), (11) and (12) into (9) we get (8).
Appendix B: Proof of Theorem 4

Let \( \{\zeta_n, n \geq 1\} \) be a sequence of independent and identically distributed (iid) Bernoulli(\( \rho \)) random variables. Denote by \((\mathcal{F}, \Omega, Pr)\) the probability space generated by this sequence. At the \( m \)-th round of DC\(^2\), if \( \zeta_m = 1 \) (which takes place with independent probability \( \rho \)) then we observe a flipped version of sign\( (x_m, h^*) \). Also, if \( \zeta_m = 0 \) we observe the correct version of sign\( (x_m, h^*) \).

Consider a query of the form sign\( (x, h^*) \). This query divides the unit circle into two parts (half-circles) depending on the sign of \( \langle x, h^* \rangle \) (see Figure 4). The two parts are: (i) Preferred part: all \( h \) such that sign\( \langle x, h \rangle = \text{sign} \langle x, h^\perp \rangle \), and (ii) Unpreferred part: all \( h \) such that sign\( \langle x, h \rangle = -\text{sign} \langle x, h^\perp \rangle \). The two parts can be separated by a line \( \ell_x \) that passes through the origin. We refer to Figure 4 for a schematic explanation.

![Preferred and Unpreferred Parts](image)

Figure 4: For any point \( z \) above the line \( \ell_x \) we have \( \langle z, h^\perp \rangle = \langle x, h^\perp \rangle \). Once we perform the query \( \langle x, h^\perp \rangle \), it is more likely that the (noisy) response is indeed the true value \( \langle x, h^\perp \rangle \). Therefore, the region above the line \( \ell_x \) is in general preferred by the query. In the figure, the sector \( (y, z) \) is cut by the line \( \ell_x \) and the sector \( (z, x) \) is not. Also, \( (z, x) \) lies in the preferred part of the query \( \langle x, h^\perp \rangle \).

In this setting, we say that the query sign\( (x, h^*) \) prefers a point \( z \) if \( z \) belongs to the preferred part of the query. Otherwise, we say that the query does not prefer \( z \). Also, we frequently use the line \( \ell_x \) rather than the query sign\( (x, h^*) \) when it causes no ambiguity. Finally, for a region \( A \) on the unit circle say that the query sign\( (x, h^*) \) cuts the region \( A \) if and only if the line \( \ell_x \) passes through region \( A \). Otherwise, we say that the query does not cut \( A \). If \( \ell_x \) does not cut \( A \), then \( \ell_x \) prefers \( A \) if \( A \) is in the preferred part and does not prefer \( A \) otherwise (see Figure 4). Finally, for two points \( x, y \) we define the distance \( d(x, y) \) to be the length of the (smaller) sector between them (see Figure 4). Clearly, we have \( d(x, y) \geq \|x - y\|^2 \).

At round \( m \) of DC\(^2\) a vector \( x_m \) is chosen and the (noisy) outcome of sign\( (x_m, h^*) \) is observed. As explained in Section ??, \( x_m \) is chosen in a way that the preferred and unpreferred parts have equal measures under \( p_{m-1} \), i.e., \( p_{m-1}(F_{x_m}) = p_{m-1}(U_{x_m}) = \frac{1}{2} \). Let us see what happens to \( p_m \) (the posterior belief about \( h^\perp \) at round \( m \)) after we conduct the query sign\( (x_m, h^*) \). As the result of the query is noisy, we have two
different update rules depending on each of the following cases: (i) \( \zeta_m = 0 \), i.e., we observe the correct value \( \text{sign}(x_m, h^*) \). In this case, the measure \( p_m \) is updated as follows:

\[
p_{m+1}(h) = \begin{cases} 
2(1 - \rho)p_m(h) & \text{if } h \in F_{x_m}, \\
(2\rho)p_m(h) & \text{if } h \in U_{x_m}.
\end{cases}
\]

(ii) \( \zeta_m = 1 \), i.e., we observe the flipped value \( -\text{sign}(x_m, h^*) \). In this case, the measure \( p_m \) is updated as follows:

\[
p_{m+1}(h) = \begin{cases} 
(2\rho)p_m(h) & \text{if } h \in F_{x_m}, \\
2(1 - \rho)p_m(h) & \text{if } h \in U_{x_m}.
\end{cases}
\]

Consider the number \( T_{\epsilon, \delta} \) given in (4). Our goal is to show that

\[
\Pr[\exists y \in S^1 : d(y, h^*) > \rho \text{ and } p_{T_{\epsilon, \delta}}(y) \geq p_{T_{\epsilon, \delta}}(h^*)] < \delta.
\]  

Clearly, the result of the theorem follows from (13). For better illustration, we assume w.l.o.g that \( h^* = (0, 1) \). Consider a point \( y \) on the right-hand side of the unit circle such that \( d(y, h^*) > \frac{\epsilon}{4} \). Also, Consider points \( z_0, z_K \) such that \( d(z_0, h^*) = \epsilon/4 \) and \( d(h^*, z_K) = \epsilon/2 \). We now divide the sector starting with \( z_0 \) and ending with \( z_K \) into \( K = T_{\epsilon, \delta} + 1 \) pints. That is, for \( i = 1, 2, \ldots, K \) we denote by \( z_i \) the point that 

\[
d(h^*, z_i) = \frac{i \epsilon}{4(T_{\epsilon, \delta} + 1)}
\]

Figure 5: Different regions for the proof of Theorem 4.

ending with \( z_i \) be denoted by \( A_i \). Note that in the very beginning of the algorithm when we have uniform measure on the unit circle, each of the regions \( A_i \) has \( p_0(A_i) = \frac{\epsilon}{8\pi(T_{\epsilon, \delta} + 1)} \) (as \( |A_i| = \frac{\epsilon}{4(T_{\epsilon, \delta} + 1)} \)).

DC\(^2\) has in total \( T_{\epsilon, \delta} \) rounds and in each round \( m \) it conducts a query with an associated line \( \ell_{x_m} \). We let \( M := \lceil \frac{2 \log \frac{\delta}{\epsilon}}{\log 4(\rho(1-\rho))} \rceil \) and consider the following events:

- \( E_1 \): There is at least \( M \) lines which separate \( z_K \) from \( h^* \) or equivalently, there is at least \( M \) lines that cut the region \( (h^*, z_K) \).

- \( E_{2,j} \) (\( 1 \leq j \leq K \)): The region \( A_j \) is not cut by any of the lines \( \ell_1, \ell_2, \ldots, \ell_{T_{\epsilon, \delta}} \).
\( E_3 : \exists y \text{ such that } d(y, h_i) > \frac{\delta}{2} \text{ and } p_{T_\rho}(y) \geq p_{T_\rho}(h_i). \)

It is easy to see that \( \Pr \left[ \bigcup_{j=1}^{K} E_{2,j} \right] = 1 \) as we have \( T_\rho \) queries and hence by the pigeon-hole principle there is always a region \( A_j \) that is not cut by any of the lines. We can write:

\[
\Pr [ E_3 ] = \Pr [ E_3 \cap E_1 ] + \Pr [ E_3 \cap E_1^c ] \\
\leq \Pr [ E_3 | E_1 ] + \sum_{j=1}^{T_\rho+1} \Pr [ E_3 \cap E_1^c \cap E_{2,j} ]. \tag{14}
\]

Now using Lemma 6 (stated below), we have

\[
\Pr [ E_3 | E_1 ] \leq \Pr [ E_3 | E_1 ] \leq (4\rho(1 - \rho))^{\frac{M}{2}} \leq \frac{\delta}{2}, \tag{15}
\]

Let us now bound \( \Pr [ E_3 \cap E_1^c \cap E_{2,j} ] \). We have

\[
\Pr [ E_3 \cap E_1^c \cap E_{2,j} ] \leq \Pr [ E_{2,j} \cap E_1^c ],
\]

and using the fact that \( |E_{2,j}| = \frac{2}{\sqrt{M_\rho T_\rho}} \) we obtain from Lemma 7 that

\[
\Pr [ E_{2,j} \cap E_1^c ] \leq (M - 1)(\theta_1 + \theta_2),
\]

and thus

\[
\sum_{j=1}^{T_\rho+1} \Pr [ E_{2,j} \cap E_1^c ] \leq (T_\rho + 1)M(\theta_1 + \theta_2), \tag{16}
\]

where \( \theta_1 \) and \( \theta_2 \) are given in Lemma 7 with \( m \leftarrow T_\rho \) and \( k \leftarrow M \). Now, we show that the above expression is upper bounded by \( \delta/2 \), and hence by using relations (14) and (15), we get the proof of the main theorem.

The value of \( T_3 \) is chosen in such a way that we have

\[
\frac{2\log(T_\rho + 1)}{T_\rho - M} \leq \frac{\log(2(1 - \rho))}{4}, \tag{17}
\]

\( T_1 \) ensures that

\[
\frac{2}{T_\rho - M} \log \left( \frac{8\pi}{\epsilon} \right) \leq \frac{\log(2(1 - \rho))}{4}, \tag{18}
\]

\( T_2 \) and ensures that

\[
\frac{2M}{T_\rho - M} \log(2\rho) \leq \frac{\log(1 - 2\rho)}{2}, \tag{19}
\]

Finally, \( T_3 \) ensures that

\[
(T_\rho + 1)M \exp \left\{ -\rho T - M \left( \frac{\log(2(1 - \rho))}{2\rho \log \frac{1}{\rho}} \right) ^2 \right\} \leq \frac{\delta}{4}, \tag{20}
\]

Now, by plugging in (17)-(20) into the values of \( \theta_1 \) and \( \theta_2 \) in (16) we conclude that the right side of (16) is bounded by \( \frac{\delta}{2} \).
Lemma 6. Let $x_1, x_2, \ldots, x_m$ be the vectors chosen by DC$^2$ up to round $m$ with $F_{x_i}$ and $U_{x_i}$ being their associated preferred and unpreferred parts (i.e. $p_{i-1}(F_{x_i}) = p_{i-1}(U_{x_i}) = 1/2$). Consider two points $h_1, h_2$ such that $h_1 \in \cap_{i=1}^m F_{x_i}$ and $h_2 \in \cap_{i=1}^m U_{x_i}$. We have for $\beta > 0$ that

$$\Pr[p_m(x) < p_m(y)] \leq (4\rho(1 - \rho))^m.$$  

Proof. For $i \in [m]$, define the random variable $Z_i$ as $Z_i \doteq \log \frac{p_i(x)}{p_i(y)}$. Using the update rules of $p_i$ that we explained above, it is easy to see that for $i \geq 1$: $Z_i = Z_{i-1} + (1 - 2\zeta_i) \log \frac{1 - 2}{\rho}$. Also, as $p_0$ is uniform over $S^1$ we have $Z_0 = 0$. We thus have $Z_m = \sum_{i=1}^m (1 - 2\zeta_i) \log \frac{1 - \rho}{\rho}$. Hence,

$$\Pr[Z_m \leq 0] = \Pr\left[\log \frac{1 - \rho}{\rho} \sum_{i=1}^m (1 - 2\zeta_i) \leq 0\right] = \Pr\left[\sum_{i=1}^m \zeta_i \geq \frac{1}{2}\right] \leq (4\rho(1 - \rho))^m,$$

where the last step follows directly from the so called Chernoff bound.

We note that the vector $h^+$ is always a member of the preferred part of any test. As a result, at any round of DC$^2$ we have that $h^+ \in \cap_{i=1}^m F_{x_i}$.

Lemma 7. Consider a region $A$ on the unit circle which does not contain $h^+$. Assume we are at round $m$ of DC$^2$ where a sequence of queries with associated lines $\ell_{x_1}, \ell_{x_2}, \ldots, \ell_{x_m}$ have been conducted. We define events $E_1$ and $E_2$ as

- $E_1 \doteq$ None of the lines $\ell_{x_i}$ cuts $A$;
- $E_2 \doteq$ At most $k$ of the lines do not prefer $A$,

where $k$ is an an integer. We have

$$\Pr[E_1 \cap E_2] \leq k(\theta_1 + \theta_2),$$

where

$$\theta_1 = \exp\left\{ -\frac{m-k}{6} \left( \frac{\log(2(1 - \rho)) - \frac{2}{m-k} \log \frac{2\pi}{|A|}}{\rho \log(\frac{1 - \rho}{\rho})} \right)^2 \right\},$$

and

$$\theta_2 = \exp\left\{ -\frac{m-k}{6} \left( \frac{\log(2(1 - \rho)) + \frac{2k}{m-k} \log(2\rho)}{\rho \log(\frac{1 - \rho}{\rho})} \right)^2 \right\}.$$
Proof. We have

\[ \Pr [E_1 \cap E_2] \leq \Pr [E_2 \mid E_1] \leq \sum_{j=1}^{k} \Pr [E_{2,j} \mid E_1], \tag{21} \]

where we define

\[ E_{2,j} \triangleq \text{Exactly } j \text{ lines do not prefer } A. \]

We will now calculate \( \Pr [E_{2,j} \mid E_1] \). In the beginning, \( p_0 \) puts a uniform measure on \( A \) and hence \( p_0(A) = \frac{|A|}{2\pi} \). Let us first investigate the dynamics of \( p_{i-1}(A) \) when we conduct the \( i \)-th query and condition on event \( E_1 \) (i.e. given that none of the lines cut \( A \)). In this setting, we define the random variables \( Z_i = \log p_i(A) \). At time \( i \), assuming that the line \( \ell_x \) does not cut \( A \), \( Z_i \) has different update rules depending on the two cases whether the line \( \ell_x \) prefers \( A \) or does not prefer \( A \). (i) first case: if the line \( \ell_x \), prefers \( A \), then we know that either with probability \( 1 - \rho \) (if \( \zeta_i = 0 \)) we have \( p_i(A) = 2(1-\rho)p_{i-1}(A) \) and with probability \( \rho \) (if \( \zeta_i = 1 \)) we have \( p_i(A) = (2\rho)p_{i-1}(A) \). Thus, we can write \( Z_i = Z_{i-1} + F_i \), where \( F_i \triangleq \zeta_i \log(2\rho) + (1-\zeta_i) \log(2(1-\rho)) \). (ii) second case: if \( \ell_x \) does not prefer \( A \), then using a similar argument we obtain \( Z_i = Z_{i-1} + U_i \), where \( U_i \triangleq \zeta_i \log(2(1-\rho)) + (1-\zeta_i) \log(2\rho) \). Now, in order to find an upper bound on \( \Pr [E_{2,j} \mid E_1] \), we assume without loss of generality that in the first \( m-j \) rounds we the lines are as in the first case and in the last \( j \) rounds the lines are as in the second case (note that any other given order of the lines is statistically equivalent to this simple order that we consider).

\[
Z_m = Z_0 + \sum_{i=1}^{m-j} F_i + \sum_{i=m-j+1}^{m} U_i = \log_2 \frac{|A|}{2\pi} + \sum_{i=1}^{m-j} F_i + \sum_{i=m-j+1}^{m} U_i.
\]

Now, noting that \( p_m(A) \leq 1 \) and hence \( \log p_m(A) \leq 0 \), we obtain

\[
\Pr [E_{2,j} \mid E_1] \\
\leq \Pr \left[ \log_2 p_0(A) + \sum_{i=1}^{m-j} F_i + \sum_{i=m-j+1}^{m} U_i \leq 0 \right] \\
= \Pr \left[ \sum_{i=1}^{m-j} F_i + \sum_{i=m-j+1}^{m} U_i \leq \log_2 \frac{2\pi}{|A|} \right]
\]

Let us now define

\[
\alpha_1 = \Pr \left[ \sum_{i=1}^{m-j} F_i \leq \log \frac{2\pi}{A} \right]
\]

and

\[
\alpha_2 = \Pr \left[ \sum_{i=1}^{m-j} F_i + \sum_{i=m-j+1}^{m} U_i \leq 0 \right]
\]

19
Using the union bound, we have
\[
\Pr [E_{2,j} \mid E_1] \leq \alpha_1 + \alpha_2.
\] (22)

Now, to bound \(\alpha_1\) we obtain after some simplifications that
\[
\alpha_1 = \Pr \left[ \sum_{i=1}^{m-j} \zeta_i \geq \rho \times \frac{m-j}{2} \times \frac{\log(2(1-\rho)) - \frac{2}{m-j} \log \frac{2\pi}{\rho}}{\rho \log \frac{1-\rho}{\rho}} \right],
\]
and by using the Chernoff bound we get
\[
\alpha_1 \leq \exp \left\{ -\rho \frac{m-j}{6} \left( \frac{\log(2(1-\rho)) - \frac{2}{m-j} \log \frac{2\pi}{\rho}}{\rho \log \frac{1-\rho}{\rho}} \right)^2 \right\}. \tag{23}
\]

To bound \(\alpha_2\) we can similarly write after some simple steps that
\[
\alpha_2 \leq \Pr \left[ \sum_{i=1}^{m-j} \zeta_i \geq \rho \times \frac{m-j}{2} \times \frac{\log(2(1-\rho)) + \frac{2j}{m-j} \log(2\rho)}{\rho \log \frac{1-\rho}{\rho}} \right],
\]
and using the Chernoff bound we get
\[
\alpha_2 \leq \exp \left\{ -\rho \frac{m-j}{6} \left( \frac{\log(2(1-\rho)) + \frac{2j}{m-j} \log(2\rho)}{\rho \log \frac{1-\rho}{\rho}} \right)^2 \right\}. \tag{24}
\]

We further note that both of the upper bounds on \(\alpha_1\) and \(\alpha_2\) decrease when we increase \(j\). Hence, the proof of the theorem follows by letting \(j = k\) in (23) and (24), and also plugging these bounds into (21).

\[
\square
\]

Appendix C: Proof of Theorem 5

First of all, we consider \(\Pr [R(\hat{h}) > \min \{ R(h_1), R(h_2) \}]\). Let \(\delta = R_\Delta(h_1) - R_\Delta(h_2)\) and \(\hat{\delta} = \hat{R}_\Delta(h_1) - \hat{R}_\Delta(h_2)\). By Hoeffding’s inequality, we have
\[
\Pr [|\hat{\delta} - \delta| \geq \gamma] \leq 2 \exp \left( -\frac{n\gamma^2}{6} \right).
\]

If \(\delta > 0\), then letting \(\gamma = \delta\) yields that \(\Pr [\hat{\delta} < 0] \leq 2 \exp(-n\delta^2/6)\). Therefore \(\Pr [\hat{R}(h_1) < \hat{R}(h_2)] \leq 2 \exp(-n\delta^2/6)\). Since we know that \(R_\Delta(h_1) > R_\Delta(h_2)\) (\(\delta > 0\)), we have \(R(h_1) > R(h_2)\) because \(h_1\) and \(h_2\) agree on the complement of \(\Delta\). Thus \(R(\hat{h}) > \min \{ R(h_1), R(h_2) \}\) with probability at most \(2 \exp(-n\delta^2/6)\). Similarly if \(\delta < 0\), then we have \(\Pr [\hat{\delta} > 0] \leq 2 \exp(-n\delta^2/6)\). Therefore \(\Pr [\hat{R}(h_1) > \hat{R}(h_2)] \leq 2 \exp(-n\delta^2/6)\). Since \(R(h_1) < R(h_2)\), hence \(R(\hat{h}) > \min \{ R(h_1), R(h_2) \}\)
with probability at most $2 \exp(-n\delta^2/6)$. In sum, we have $R(\hat{h}) > \min\{R(h_1), R(h_2)\}$ with probability at most $2 \exp(-n|\Delta(h_1) - \Delta(h_2)|^2/6)$.

Let $\delta_n$ denote $2 \exp(-n|\Delta(h_1) - \Delta(h_2)|^2/6)$. Now we want to bound $\mathbb{E}[R(\hat{h}) | h_1, h_2]$. We have

$$\mathbb{E}[R(\hat{h}) | h_1, h_2] \leq (1 - \delta_n) \min\{R(h_1), R(h_2)\} + \delta_n \max\{R(h_1), R(h_2)\}$$

$$= \min\{R(h_1), R(h_2)\} + \delta_n |R(h_1) - R(h_2)|$$

$$\leq \min\{R(h_1), R(h_2)\} + 2|R(h_1) - R(h_2)|$$

$$\leq \min\{R(h_1), R(h_2)\} + 2|\Delta(h_1) - \Delta(h_2)|$$

$$\leq \min\{R(h_1), R(h_2)\} + 2|\Delta(h_1) - \Delta(h_2)|$$

where the last inequality holds because $|R(h_1) - R(h_2)| \leq |\Delta(h_1) - \Delta(h_2)|$.

When $u = \sqrt{3/n}$, the function $f(u) = 2ue^{-nu^2/6}$ achieves its maximum $2\sqrt{\frac{3}{ne}}$. Thus we obtain that

$$\mathbb{E}[R(\hat{h}) | h_1, h_2] \leq \min\{R(h_1), R(h_2)\} + 2\sqrt{\frac{3}{ne}}.$$