How Close is your Function to Depending on a Small Number of its Inputs?

Michael Whitmeyer

Acknowledgement

I would like to thank Avishay for his guidance, help, insights, and kindness whenever I was stuck. I have learned an incredible amount just by witnessing his ideas and approaches to complex topics in real time. Without his guidance, this thesis would not have been possible. I would also like to thank Vishnu for always being available for to discuss all manner of dumb ideas. I would not like to thank Covid-19. Go Bears.

(SAMPLE)

How Close is Your Function to Depending on a Small Number of its Inputs?

by Michael Whitmeyer

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

[illegible]

Professor Avishay Tal
Research Advisor

May 21, 2021

(Date)

* * * * * * *

[illegible]

Professor Prasad Raghavendra
Second Reader

(S A M P L E)

____

May 14th, 2021

How Close is Your Function to Depending on a Small Number of its Inputs?

by
Michael Whitmeyer

A thesis submitted in partial satisfaction of the requirements for the degree of Master of Science in Computer Science in the Graduate Division of the University of California, Berkeley

Spring 2021

How Close is Your Function to Depending on a Small Number of its Inputs?

Abstract

This thesis focuses on the following question: if we have query access to some $f : \{-1,1\}^n \to \{-1,1\}$ (meaning we can query any of its 2^n outputs at will), how many queries do we need to discern whether f is "close" or "far" from depending on only a subset of k of its inputs? Such functions are known in the field as "k-juntas". The main results come from the recent work of [ITW21], which gave an improved upper bound on the query complexity of this problem. Namely, it was shown that only $2^{\widetilde{O}(\sqrt{k})}$ queries to f suffice to answer this question. Along the way, we will provide a more detailed and leisurely survey of the other results and progress in this area, examining other techniques and the strongest known upper and lower bounds for variants of this problem.

To my parents.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank Avishay for his guidance, help, insights, and kindness whenever I was stuck. I have learned an incredible amount just by witnessing his ideas and approaches to complex topics in real time. Without his guidance, this thesis would not have been possible. I would also like to thank Vishnu for always being available for to discuss all manner of dumb ideas. I would not like to thank Covid-19. Go Bears.

Chapter 1

Survey of Junta Testing and Variants

1.1 Property Testing

The study of property testing, initiated by Blum, Luby, and Rubinfeld in their seminal work on linearity testing [BLR90], is concerned with making fast decisions about a global object having some global property, while only accessing (or "querying") parts of it.

We begin our discussion with a canonical example, which was the original problem considered in [BLR90]. They wanted to decide whether a given Boolean function $f : \{0,1\}^n \to \{0,1\}$ was *linear*, meaning $f(x+y) = f(x) + f(y)$ for all $x, y \in \{0,1\}^n$.[1] This is equivalent to testing whether $f = \sum_{i \in S} x_i$ for some subset $S \subseteq [n]$,[2] or whether it is ε-far from *every* linear function. Here, we must define what we mean when we say f and g are "ε-far". To discuss this example, we need the following notion of distance.

Definition 1.1.1. *We define the* distance *of two Boolean functions $f, g : \{0,1\}^n \to \{0,1\}$ to be the fraction of inputs on which they disagree:*

$$\mathsf{dist}(f,g) := \frac{|\{x \mid f(x) \neq g(x)\}|}{2^n} = \mathbf{Pr}[f(x) \neq g(x)].$$

*Moreover, we can extend this notion to a class of functions $\mathcal{C}$. We say f is ε-*close *to $\mathcal{C}$ if there exists some $g \in \mathcal{C}$ such that $\mathsf{dist}(f,g) \leq \varepsilon$, otherwise we say f is ε-*far *from $\mathcal{C}$. More generally, $\mathsf{dist}(f,\mathcal{C}) := \min_{g \in \mathcal{C}} \mathsf{dist}(f,g)$.*

We can now return to the above discussion of linearity testers. Here, $\mathcal{C}$ is the family of all linear functions. Perhaps surprisingly, [BLR90] was able to show that for any constant ε, only a constant number of queries suffices to distinguish between the case that $f \in \mathcal{C}$ and $\mathsf{dist}(f,\mathcal{C}) > \varepsilon$. In particular, the following was their "test" for linearity (which is perhaps the first test you might consider trying!):

1. Choose random $x \in \{0,1\}^n$ and $y \in \{0,1\}^n$.

2. Query f at x, y, and $x+y$.

[1] Here $x+y$ denotes coordinate-wise vector addition mod 2.

[2] See [O'D14], exercise 1.26 for more on this equivalence.

3. Accept if $f(x) + f(y) = f(x + y)$.

It is clear that if f is actually linear, this test will always accept. The nice result is that if f is ε-far from linear, then this test accepts with probability at most $1 - \varepsilon$! A particularly clean analysis for this fact can be found in O'Donnell [O'D14], Chapter 1.6.

The above is a favorite seminal example, and since then property testing was further explored by Goldreich, Goldwasser, and Ron [GGR96], who drew connections to the areas of learning theory and approximation algorithms in the context of graph properties. Even before that, however, many were interested in property testing in the context of polynomials and program checking, since low-degree tests are a crucial part of the proofs that $\mathsf{MIP} = \mathsf{NEXP}$ [BFL91] and $\mathsf{NP} = \mathsf{PCP}[\log n, O(1)]$ [ALM$^+$92]. The latter "PCP Theorem" of course being of great interest due to its many applications in hardness of approximation results (see e.g. [Tre04] for a survey).

By now, property testing by itself is an extremely well-established field, and indeed at least one textbook has been written on the topic by Goldreich [Gol17]. In this thesis we focus on properties of Boolean functions. For convenience, we will represent Boolean functions primarily as $f : \{\pm 1\}^n \to \{\pm 1\}$.[3]

We are now ready to state the definition, which has already been alluded to, of a property testing algorithm $\mathcal{A}$. Given $\varepsilon > 0$ and a class of functions $\mathcal{C}$, we say that the algorithm $\mathcal{A}$ is a property tester for $\mathcal{C}$ if it satisfies the following two conditions:

1. if $f \in \mathcal{C}$, then $\mathcal{A}$ accepts f with probability at least $2/3$;

2. if $\mathsf{dist}(f, g) \geq \varepsilon$ for all $g \in \mathcal{C}$, then $\mathcal{A}$ rejects with probability at least $2/3$.[4]

The primary measure of efficiency for such property testing algorithms is the algorithm's *query complexity*, or the number of times it must use its black box access to f. Such query algorithms can be *adaptive* in that the coordinates on which they query f depend on previous answers, or they can be *nonadaptive* in that the algorithm always queries f in a predetermined manner. In this thesis we survey both adapative and nonadaptive upper and lower bounds, but the main result presented will be an adaptive algorithm.

1.1.1 Tolerant Property Testing

One of the first relaxations of the standard property testing model considered (sometimes referred to as the "parameterized" regime) were testers that distinguished between $f \in H$ and f being ε-far from $H' \supseteq H$. This notion was introduced by Kearns and Ron [KR00] in the context of testing decision trees and certain classes of neural networks. We note that if H' is a strict superset of H, then the job of the tester becomes easier, and smaller query or sample complexity is often achievable than in the regular testing model. Indeed, our Theorem 1.8.2 is an example of a (tolerant) parameterized tester.

Tolerant testing is yet another generalization of the standard property testing model. The notion was first introduced by Parnas, Ron, and Rubinfeld [PRR04]. Standard property

[3]This is largely because Fourier analysis of Boolean functions becomes easier to work with when Boolean functions are represented in this way. In this basis, $0 \mapsto 1$ and $1 \mapsto -1$, and addition mod 2 corresponds to multiplication.

[4]As is common for such probabilistic decision problems, the choice of $2/3$ here is arbitrary. As long as these probabilities are bounded away from $1/2$, we can repeat the algorithm and take the majority vote of all the outcomes (here, Chernoff/Hoeffding bounds are our friends) to achieve any accept/reject probabilities that we may desire.

testing entails distinguishing between functions that *exactly* satisfy a certain property, and functions that are ε-far from satisfying said property. This is somewhat restrictive, and the tolerant testing problem seeks to more generally distinguish functions that are c_ℓ close to having the desired property, and those that are at least c_u far from having the property, for some $0 < c_\ell < c_u < 1$. We also note that the notion of tolerant testing is closely related to the notion of distance approximation – indeed, if one can estimate $\mathsf{dist}(f, \mathcal{C})$ up to additive error $(c_u - c_\ell)/2$ with probability at least $2/3$, then one has solved the tolerant testing problem for that class.[5] In general, tolerant testing (and therefore distance approximation), is much more challenging than traditional property testing. Figure 1.1 provides a visualization of the tolerant testing problem. Tolerant testing has received a lot of attention recently, see for example [BLT20] for work on tolerant testing of decision trees and [ACCL07] [PRW20] for work on tolerant testing of monotonicity. This thesis focuses primarily on the tolerant testing of juntas, which we discuss extensively in Section 1.7.

1.2 Motivating Junta Testing

First and foremost, we must mention the survey by Blais [Bla10] which gave a nice overview of the state of nontolerant junta testing at the time, which was 2010. In this section, we go over many of the same results, but also present more recent work in the area and improved upper and lower bounds.

In this thesis, we will focus primarily on the *tolerant* testing of *k-juntas*, which are the class of Boolean functions on n variables which depend on only k of their inputs.

To those outside the field of property testing, the focus on juntas may at first seem a bit puzzling. Indeed, why even further restrict ourselves to only Boolean functions? Before we defend why juntas and Boolean functions matter so much, we must mention that much effort has also been put into testing other properties, such as monotonicity [PRW19, ACCL07], graph properties [GGR96] (see [Gol11] for a survey), and distribution testing (see [Can15] for a survey), just to name a few.

Why, then, are juntas on Boolean functions so interesting[6]? First, there are nice connections to machine learning and the learning theory community in general cares about them a lot [GE03, Val12, MOS03]. Data is getting bigger and bigger, and as that happens it becomes more and more likely that any function of said data actually depends on only a small subset of it. Those are juntas! A favorite canonical example comes from genetics – for example, is a disease or phenotype a function of a few genes, or far from it? If you knew your function was a junta (or at least close), then you could in principle run a more learning algorithm to figure out exactly *which* junta you were close to, rather than just testing it. This would give you more information and arguably be more useful in the real world, however, such learning algorithms are expensive.[7] Therefore, it may be prudent to first *test* that your function is close to a k-junta (which is cheap) before running the expensive learning algorithm. Another nice thing about juntas is that they naturally

[5]The reverse direction is also true up to logarithmic factors in the query complexity – given a tolerant tester it is possible to estimate the distance to that property. See for example section 3 in [ACCL07] and section 5 in [PRW19].

[6]to me

[7]Learning k-juntas on n Boolean variables must information-theoretically depend on the ambient dimension n, and indeed the state of the art k-junta learners don't do terribly much better than the trivial $\binom{n}{k}\mathsf{poly}(2^k, n)$ algorithm one can achieve by simply iterating over all possible subsets of size k and seeing how well they work [MOS03, Val12].

capture a lot of other types of functions. For example, dictators/antidictators (functions $f : \{0,1\}^n \to n$ of the form $f(x) = x_i$ or $f(x) = 1 - x_i$), DNFs, CNFs, decision trees, k-linear functions ($f(x) = \sum_{i \in S} x_i \mod 2$, k-monomials (functions $f : \{0,1\}^n \to n$ of the form $f(x) = \prod_{i \in S} x_i$), and low degree polynomials are all naturally some form of juntas. This makes juntas a very general class to work with, although it is sometimes possible to have more efficient learning/testing algorithms for more specific kinds of juntas.

1.3 Preliminaries

Before we begin our survey, we must discuss some basic preliminaries concerning Boolean functions. For the mjority of this thesis, we will represent Boolean functions primarily using the multiplicative group $\{\pm 1\}$, that is, $f : \{\pm 1\}^n \to \{\pm 1\}$. Occasionally, when it is more convenient, we represent Boolean functions over $\mathbb{Z}_2 = \{0,1\}$. The distinction is not very consequential, it just requires translating between multiplication and addition modulo 2 (both are XOR's).

We denote the *complement* $\bar{f}$ to be $\bar{f}(x) := -f(x)$ (if we were working in the $\{0,1\}$ basis, $\bar{f}(x) := 1 - f(x)$). All addition and multiplication of vectors on $\{0,1\}^n$ will be taken pointwise. For a subset $S \subseteq [n]$, the vector $z = x_S y_{\bar{S}}$ is formed by taking $z_i = x_i$ for all $i \in S$, and $z_i = y_i$ for all $i \notin S$.

For a positive integer n, we denote by $[n]$ the set $\{1, \ldots, n\}$. For a distribution $\mathcal{D}$, we denote that a random variable x is sampled according to $\mathcal{D}$ by $x \sim \mathcal{D}$. In the case that x is sampled uniformly at random from a set S, we will abuse notation slightly and write $x \sim S$.

The binomial distribution with n trials and probability p per trial will be denoted $\mathrm{Bin}(n,p)$. We denote the set $\{-1,1\}$ with the shorthand $\{\pm 1\}$. For functions f, g from $\{\pm 1\}^n$ to $\{\pm 1\}$ we define $\mathsf{dist}(f,g) = \mathbf{Pr}_{x \sim \{\pm 1\}^n}[f(x) \neq g(x)]$: that is, the fraction of inputs on which f and g differ. For a set $S \subseteq [n]$ we will denote by $\{\pm 1\}^S$ the set of possible assignments to the variables $\{x_i\}_{i \in S}$.

1.3.1 Fourier Analysis of Boolean Functions

In this subsection we recall some tools in the analysis of Boolean functions. For a more thorough introduction to the field, we refer the reader to [O'D14]. For every subset $S \subseteq [n]$, we define the parity function on the bits in S, denoted by $\chi_S : \{\pm 1\}^n \to \{\pm 1\}$ as $\chi_S(x) = \prod_{i \in S} x_i$. It is a well-known fact that we can express uniquely any $f : \{\pm 1\}^n \to \mathbb{R}$ as a linear combination of $\{\chi_S\}_{S \subseteq [n]}$:

$$f(x) = \sum_{S \subseteq [n]} \widehat{f}(S) \chi_S(x).$$

The coefficients $\{\widehat{f}(S)\}_{S \subseteq [n]}$ are referred to as the Fourier coefficients of f, and can be calculated by $\widehat{f}(S) = \mathbf{E}[f(x)\chi_S(x)]$. We say Fourier coefficients are on *level* s if they correspond to subsets of size s.

Definition 1.3.1 (Influence of Coordinates and Sets). *We define the* influence *of coordinate i on $f : \{0,1\}^n \to \{0,1\}$ as*

$$\mathbf{Inf}_i[f] := \mathbf{Pr}[f(x) \neq f(x^i)],$$

where the probability is taken over the uniform distribution and x^i denotes the input x with the i^{th} coordinate flipped. It is a well-known fact (see, e.g., [O'D14, Theorem 2.20]) that

$\mathbf{Inf}_i[f] = \sum_{S \ni i} \widehat{f}(S)^2$. *We also naturally define, following the convention of [Bla09, Bla08, BCE+19], the influence of a set S on f as*

$$\mathbf{Inf}_S[f] = 2 \cdot \Pr_{x,y}[f(x) \neq f(x_{\bar{S}}, y_S)].$$

We naturally extend the above definition and define the *low-degree influence* (up to level k) of coordinate i on f as

$$\mathbf{Inf}_i^{\leq k}[f] = \sum_{S \ni i, |S| \leq k} \widehat{f}(S)^2.$$

For a set $T \subseteq [n]$ we define the *projection* of the function f to T, denoted $f^{\subseteq T}$, as the partial Fourier expansion restricted to sets contained in T, i.e., $f^{\subseteq T}(x) = \sum_{S: S \subseteq T} \widehat{f}(S) \chi_S(x)$. We observe that $f^{\subseteq T}$ depends only on coordinates in T and that it can be alternatively defined as $f^{\subseteq T}(x) = \mathbf{E}_{y \sim \{\pm 1\}^n}[f(y) | y_T = x_T]$. As suggested by the last identity, we also denote $f^{\subseteq T}$ by $f_{\mathsf{avg},T}$.

In the regime of property testing, we define the "correlation" of two functions (which is related the the distance between two functions) as follows.

Definition 1.3.2 (Correlation). *For functions $f, g : \{\pm 1\}^n \to \{\pm 1\}$ and a family of functions G, all from $\{\pm 1\}^n \to \{\pm 1\}$ we say that*

1. *f and g are c-correlated if $\mathbf{E}_{x \in \{\pm 1\}^n}[f(x)g(x)] = c$;*
2. *f and G are c-correlated (denoted $\mathsf{corr}(f, G) = c$) if $\max_{g \in G} \mathbf{E}_{x \in \{\pm 1\}^n}[f(x)g(x)] = c$.*

In the paper, we will occasionally abbreviate the correlation between f and g as $\mathbf{E}[fg]$ when the domain is implied. Observe that when f and g are Boolean-valued (in ± 1) we have $\mathbf{E}[fg] = 1 - 2\mathsf{dist}(f, g)$.

Fact 1.3.3. *For functions $f, g : \{\pm 1\}^n \to \mathbb{R}$, we have Plancheral's identity:*

$$\mathop{\mathbf{E}}_{x \sim \{\pm 1\}^n}[f(x)g(x)] = \sum_{S \subseteq [n]} \widehat{f}(S)\widehat{g}(S) \ .$$

When $f = g$, this fact is known as Parseval's identity.

Now, we define some classes of Boolean functions with properties that will be useful to us.

Definition 1.3.4 (Junta). *Let $T \subseteq [n]$. A function $f : \{\pm 1\}^n \to \mathbb{R}$ is called a* junta *on T if f depends only on coordinates in T. I.e., there exists a function $g : \{\pm 1\}^T \to \mathbb{R}$ such that $f(x) = g(x_T)$. A function is called a k-*junta *if it is a junta on T for some $T \subseteq [n]$ of size k. Following the notation of [DMN19], we denote the class of k-juntas on n inputs as $\mathcal{J}_{n,k}$. We also denote $\mathcal{J}_{U,k}$ as the set of k-juntas with inputs inside of U, and when $|U| = k$ then we often denote $\mathcal{J}_U := \mathcal{J}_{U,k}$ for brevity.*

1.4 Testing 1-Juntas

We start with an examination of the simplest form of junta, the set of functions that depend on at most one variable. Since we are dealing with Boolean functions, the family of such functions is remarkably small (when considering that on n bits, there are 2^{2^n} possible functions). It only contains dictators, antidictators, and constant functions, so in total our family contains $2n+2$ functions. We present the following algorithm, the core ideas of which come from [BLR90, BCH+96]:

Algorithm 1: Dictator/Constant Test

```
Input: $f : \{0,1\}^n \to \{0,1\}$ (target function)
Output: "Accept" w.p. 2/3 if $f$ is a dictator or constant function, otherwise
        "Reject" w.p. 2/3 if $f$ is $\varepsilon$-far from all such functions.
1 for $O(1/\varepsilon)$ random pairs $x, y \in \{0,1\}^n$ do
      1. Verify that $f(x) + f(y) = f(x+y)$ (otherwise reject) /* verify linearity */
      2. Verify that $f(x) \cdot f(y) = f(x \cdot y)$ (otherwise reject) /* verify monomiality */
2 Accept if all tests pass.
```

Intuitively, Algorithm 1 checks that f is both linear and a monomial. The only such functions for which that is true are dictators and constant functions. Since anti-dictators are also included in the family of 1-juntas, we would also like our test to account for this. This is a simple fix: we can run Algorithm 1 twice, once with f and another time with $\bar{f}$, and accept if it accepts at least once. We already established in Section 1.1 that the linearity test works properly – it remains to note that the monomiality test will reject functions that are parities of at least two variables:

Lemma 1.4.1 ([PRS01, BGS98])**.** *The probability that Item 2 in Line 1 of Algorithm 1 passes for* $f = \sum_{i \in S} x_i \mod 2$ *is equal to*

$$\begin{cases} 1/2 + 2^{-|S|-1} & \text{if } |S| \text{ is even} \\ 1/2 + 2^{-|S|} & \text{if } |S| \text{ is odd.} \end{cases}$$

For the proof of this lemma, we refer the reader to the original work, but it is a relatively simple counting argument. With this and the fact that the linearity testing part of Algorithm 1 works, this provides us all the analysis we need to say that we have achieved a 1-Junta tester with query complexity $O(1/\varepsilon)$. We note however that this test is somewhat "old-fashioned" in that there are newer and perhaps even simpler dictator/antidictator tests that can be found in Chapter 7 of [O'D14], for the curious reader.

1.5 A Brief History of Junta Testing and its Variants

Now that we have seen a simple tester for 1-juntas, it is time to move to the more challenging task of testing k-juntas. The main goal of k-junta testers is to have a query complexity that is independent of the ambient dimension n.[8]

At first glance, this might seem hopeless, but Fischer et al. [FKR+04] were the first to come up with a clever way to "get rid of" the dependence of their tester on n. Their insight/intuition was to *partition* the n inputs of f into $\mathsf{poly}(k)$ many "buckets". The very high level idea is that naturally, if f is a k-junta, then it is certainly a "k-part" junta with respect to this partition into buckets (i.e. it is only dependent on at most k parts). The

[8] Note that it is not possible to have a runtime that is independent of n, since even just making a query requires specifying the n bits of the input to the oracle.

more nontrivial thing one most prove is that if f is ε-far from any k-junta, then it is also, with high probability over the random partition of the inputs, far from a k part junta. We formalize this more soon, but there is yet another key ingredient, which was actually first an observation of Blum, Hellerstein, and Littlestone [BHL95], but which was used to great effect by Blais [Bla09]. The ingredient is the following observation: if we have two inputs x and y such that $f(x) \neq f(y)$, then at least one of the coordinates i where $x_i \neq y_i$ must be relevant (have nonzero influence). Moreover, by performing a simple binary search over the hybrid inputs between x and y,[9] we can *find* such a relevant coordinate with only $\log n$ queries. Crucially, the exact analogous procedure works to find a relevant *part* of f with only $O(\log k)$ queries if we have partitioned the inputs of f into $\mathsf{poly}(k)$ parts/buckets.

Now that we have some high level ideas, we can present some more of the details of these algorithms.

One can alternatively define a k-junta as a function f that contains a set S of size at most k such that $\mathbf{Inf}_{\bar{S}}[f] = 0$. Also, the above definition suggests an extremely natural way of estimating the influence of a set: simply sample $x, y \sim \{0,1\}^n$ and then check if $f(x) = f(x_{\bar{S}} y_S)$. If the set has influence γ, then repeating this $O(1/\gamma)$ suffices to see at least one such pair (x, y) such that the check fails with high probability, and we can be confident f depends on at least one coordinate in S. This leads to the following algorithm found in [FKR+04]:

Algorithm 2: Fischer et al.'s [FKR+04] k-junta test

```
Input: $f : \{0,1\}^n \to \{0,1\}$ (target function)
Output: "Accept" if $f$ is a $k$-junta with probability 2/3, and "Reject" if $f$ is $\varepsilon$-far
        from any $k$-junta with probability 2/3.
1 Randomly partition the coordinates into $O(k^2)$ buckets $S_1, ..., S_{O(k^2)}$.
2 for each $S_i$ do
3     Check that $f$ is independent of $S_i$ by sampling $O(k^2/\varepsilon)$ pairs $x, y \sim \{0,1\}^n$ and
      verifying that $f(x) = f(x_{\bar{S}_i} y_{S_i})$.
4 return "Accept" if at most $k$ such $S_i$ fail, otherwise "Reject"
```

[FKR+04] then proved the following lemma (using Fourier analysis of Boolean functions) to verify the soundness of the above procedure.

Lemma 1.5.1 ([FKR+04])**.** *If f is ε-far from every k-junta, then with high probability over a random partition $S_1, ..., S_{O(k^2)}$ will have at least $k+1$ parts with influence $\mathbf{Inf}_S[f] > \varepsilon/k^2$.*

The above lemma directly implies the soundness of Algorithm 2, and implies a $O(k^4/\varepsilon)$-query junta test, which is notably independent of n. This represented the first major breakthrough in junta testing, and in the same paper a slightly more clever (adaptive) algorithm that achieved $\widetilde{O}(k^2)$ queries was presented. However these algorithms did not use the aforementioned observation of Blum et al. and Blais (which was really just a clever application of binary search). So how does this help exactly? We are now ready to present

[9] In more detail: let S be the set of coordinates on which x and y differ. Then we can flip the bits of x for half of the coordinates in S, call this input z. Then either $f(x) \neq f(z)$ or $f(z) \neq f(y)$. Whichever is the case, we can recurse on that pair of inputs, and this takes $\log n$ time and queries.

the optimal[10] adaptive algorithm of Blais [Bla09] in Algorithm 3.

Algorithm 3: Blais's [Bla09] optimal k-junta test.

```
Input: $f : \{0,1\}^n \to \{0,1\}$ (target function)
Output: "Accept" if $f$ is a $k$-junta with probability 2/3, and "Reject" if $f$ is $\varepsilon$-far
        from any $k$-junta with probability 2/3.
1 Randomly partition the coordinates into $O(k^2)$ buckets $S_1, ..., S_{O(k^2)}$.
2 Initialize $J = \emptyset$. /* Start with no relevant parts of $f$ */
3 for $O(k/\varepsilon)$ rounds do
4     Set $J \leftarrow J \cup$ Find New Relevant Part$(f, \mathcal{I}, \bar{J})$. /* Use Algorithm 4 to find
          new relevant part. */
5     if $J$ contains $> k$ relevant parts then
6         return "Reject"
7 return "Accept"
```

Algorithm 4: Find New Relevant Part

```
Input: $f$, Partition $\mathcal{I}$ of $[n]$, subset $S \subseteq [n]$
Output: A bucket of the partition (in $S$) that contains a relevant variable to $f$,
        otherwise outputs the empty set
1 Sample $x, y \sim \{0,1\}^n$ u.a.r.
2 if $f(x) \neq f(x_{\bar{S}} y_S)$ then
3     use binary search to find a part $I \in \mathcal{I}$ (in $S$) with a relevant variable.
4     return $I$
5 Otherwise, return $\emptyset$
```

We must note that Algorithm 3 is slightly different than originally presented in [Bla09], wherein Blais partitioned the inputs into buckets of size $O(k^9/\varepsilon^5)$. Blais in [Bla09] also proved his result using the Efron-Stein decomposition of functions, which extends Fourier/harmonic analysis to more general functions with finite product domains with finite ranges. A somewhat simpler algorithm/analysis for the case of Boolean functions, wherein Blais partitions into $O(k^2)$ buckets as we have presented in Algorithm 3, can be found in Blais's PhD thesis [Bla12].

What has to happen in order for this algorithm to work? Once again, it is clear that if f is a k-junta, then it is certainly a k-part junta, so it just remains to show that if f is ε-far then Algorithm 3 will reject with high probability. Soundness follows from the following lemma.

Lemma 1.5.2 (Lemma 5.4 in [Bla12])**.** *If $f : \{0,1\}^n \to \{0,1\}$ is ε-far from any k-junta, and $\mathcal{I}$ partitions $[n]$ into $O(k^2)$ buckets, then with high probability every set J formed by taking the union of at most k parts of $\mathcal{I}$ satisfies $\mathbf{Inf}_{\bar{J}}[f] \geq \varepsilon/2$.*

With this lemma in hand, soundness of Algorithm 3 is easy to show, and it can also be easily verified that the algorithm makes at most $O(k \log k + k/\varepsilon)$ queries. Finally, we note that [Bla08] gives a $\widetilde{O}(k^{3/2})$ query junta tester, which is slightly worse in complexity but crucially does not require any adaptive queries.

We now mention some a few generalizations (namely, the distribution free model and the quantum model) of the property testing model and the work that has been done in these

[10]We will get to why it's optimal (for constant ε) shortly.

models on the problem of junta testing.

1.5.1 Distribution Free Setting

Every algorithm and definition we have mentioned thus far have operated under the assumption that the distribution over the inputs of our function is uniform. It is natural to consider what happens when this is not the case. This model of property testing is known as the *distribution free* model, and was first considered in the context of property testing by Halevy and Kushilevitz [HK07]. For the distribution free testing of juntas, Chen et al. [LCS$^+$19] initially gave a $\widetilde{O}(k^2/\varepsilon)$-query algorithm with one-sided error. Notably, their algorithm is adaptive, and they were able to show that adaptivity is crucial in the distribution-free setting by exhibiting a $\Omega(2^{k/3})$ query lower bound for nonadaptively testing juntas in the distribution free model. This was quickly followed up by the works of Bshouty [Bsh19b] and Zhang [Zha19] who gave $\widetilde{O}(k/\varepsilon)$-query algorithms with two-sided and one-sided error, respectively. The methods utilized by Bshouty extend those of Diakonikolas et al. [DLM$^+$07] and result in algorithms not only for junta testing but also several subclasses of juntas.

As a precursor of what is to come in Chapter 2, which presents the recent result [ITW21], we note that while we solve a similar problem in a different setting,[11] some of our techniques resemble those of [Bsh19b]: notably, an idea introduced in [Bsh19b] is to find a witness such that, if all coordinates outside a subset of the coordinates are fixed to this witness' values, then f becomes a dictator on a single coordinate within that subset. This can be thought of as obtaining oracle access to a relevant coordinate, an idea pervasive throughout the work of [DMN19] and ours. The techniques in [DLM$^+$07, DLM$^+$08, Bsh19b] can all be categorized in the "testing via implicit learning" paradigm, as surveyed in [Ser10].

1.5.2 Quantum Junta Testing

In a seminal paper, Beals et al. [BBC$^+$01] introduced the *quantum* oracle model, wherein one not only has query access to f, but can query f on *superpositions* of inputs, and receive superpositions over the outputs. Clearly this model is strictly more powerful than the classical oracle model, since one can always just not query in superposition. The quantum analog of property testing was subsequently defined and examined by Buhrman et al. [BFNR08], in which they exhibited several functions with good quantum testers, yet no efficient classical testers.

The natural question for us of course, is if the power of querying in superposition helps for the specific problem of junta testing. This was first explored by Atici and Servedio [AS07], who exhibited an $O(k/\varepsilon)$ quantum query algorithm for k-junta testing.[12] This actually already showed that the quantum model provided strictly more power than the classical model (by a logarithmic factor) due to the lower bound of Sağlam [Sag18] (which we discuss more in the next section), but this was not yet known at the time. [AS07] also gave a lower bound of $\Omega(\sqrt{k})$ for k-junta testing in this model. Subsequently, Ambainis et al. [ABRdW16] gave an improved quantum k-junta tester that used only $\widetilde{O}(\sqrt{k/\varepsilon})$ quantum

[11] As a reminder, the results in Chapter 2 pertain to the *tolerant* testing of juntas in the uniform distribution model.

[12] They also had several results for the *learning* of k-juntas in this model (that, crucially, had query complexity independent of n), and their work also built off of the previous work of Bshouty and Jackson [BJ99], which demonstrated how to sample according to the Fourier spectrum of f in the quantum query model.

queries to f, which by the lower bound of [AS07] is optimal up to logarithmic factors for constant ε.

1.5.3 Summary

Throughout this section we have discussed several algorithms and variants of junta testing. Table 1.1 summarizes this information.

Reference	Model	Adaptive?	Query Complexity
[FKR+04]	Classical	No	$\widetilde{O}(k^2)/\varepsilon$
[Bla08]	Classical	No	$\widetilde{O}(k^{3/2})/\varepsilon$
[Bla09]	Classical	Yes	$O(k \log k + k/\varepsilon)$
[LCS+19]	Distribution Free	Yes	$\widetilde{O}(k^2)/\varepsilon$
[Bsh19b, Bsh19a, Zha19]	Distribution Free	Yes	$\widetilde{O}(k/\varepsilon)$
[AS07]	Quantum	No	$O(k/\varepsilon)$
[ABRdW16]	Quantum	Yes	$\widetilde{O}(\sqrt{k/\varepsilon})$

Table 1.1: Table of Nontolerant Junta Testers

We remark that in Table 1.1, the results of Zhang [Zha19] and Bshouty [Bsh19b] are quantitatively similar but use different techniques, and Zhang's algorithm has only one sided error.

1.6 Lower Bounds for Junta Testing

We have seen many algorithmic results in the previous sections for nontolerant junta testing. As theoreticians, we always want to understand if we can do better, or conversely, if we can prove that we cannot do better. The focus of this section is thus on surveying the work that has been in the context of lower bounding the complexity of nontolerant junta testing. Luckily, the field of property testing falls within the realm of the query model, which is a relatively nice model to work in for lower bounds.

1.6.1 Adaptive Lower Bounds

The first lower bounds came along with the first (nontrivial) upper bounds in the paper of Fischer et al. [FKR+04]. They showed that in order to test k-juntas, one must query the function in at least $\widetilde{\Omega}(\sqrt{k})$ times. This lower bound was for nonadaptive algorithms, but we mention it here since it was the first nontrivial result. We note a couple things about this lower bound. First, it implied an $\Omega(\log k)$ lower bound for adaptive algorithms. This is because any adaptive algorithm can be made nonadaptive by simply enumerating over all the possible answers of the q queries the adaptive algorithm makes, and querying what the adaptive algorithm would've having seen those answers. This results in a at most 2^q-query nonadaptive algorithm (so more generally, a 2^q lower bound for nonadaptive algorithm implies a q lower bound for adaptive algorithms). Second, we note that the lower bound of [FKR+04] only applied to the case when $k = o(\sqrt{n})$, although this is not a wholly unreasonable assumption to make, since we usually consider $k \ll n$ (otherwise there is no need to go through all this trouble eliminating the dependency of the query complexity on n).

[FKR$^+$04] proved their result using Yao's minimax principle. We discuss this principle, and its derivation, extensively in Section A. At a high level, Yao's principle states that the complexity of the best randomized algorithm on the worst input corresponds to the complexity of the best deterministic algorithm on the hardest possible input distribution, and can be derived from the more general minimax theorem. This allows one to lower bound the complexity of a randomized algorithm by defining a "hard" distribution over inputs and then analyzing a deterministic algorithm over such inputs (analyzing deterministic algorithms over distributions of inputs is often much easier than analyzing randomized algorithm, hence why Yao's principle is used so much in practice).

The lower bound of [FKR$^+$04], along with their $\widetilde{O}(k^2)$ upper bound, left a quartic gap between the upper and lower bounds at the time, but these were nonetheless huge breakthroughs. We also note that the result was proven (roughly) by defining input distributions over parity functions of size k (the "yes" instances) and parity functions of size $k + 2$ (the "no" instances), and then using a random walk argument to bound the statistical distance between these two distributions. Shortly after [FKR$^+$04], Chockler and Gutfreund [CG04] gave an improved lower bound of $\Omega(k)$, and as an added bonus their lower bound worked even for adaptive algorithms. They also used Yao's principle for their lower bound, and their distribution was simply over completely random functions either depending on k inputs (the yes case) or $k + 1$ inputs (the no case). Their analysis was also much simpler, using only elementary arguments and inequalities. Moreover, it remained the best known lower bound for this problem for about 15 years. This result of Chockler and Gutfreund [CG04] is the reason we said the adaptive algorithm of Blais [PRR04] was nearly optimal – the upper and lower bounds only differ by a logarithmic factor (for constant ε).

In 2012, Blais, Brody, and Matulef [BBM12] provided the next breakthrough in the realm of property testing lower bounds by showing that it was possible (via relatively simple reductions) to translate communication complexity lower bounds into property testing lower bounds. For the unfamiliar reader, communication complexity is a model in which two parties, usually called Alice and Bob, are trying to compute a function (of their respective inputs) by communicating as little as possible. We shall not dwell on the details/nuances of the communication model in this thesis, but for the curious reader we point to two textbooks on the topic, one by Kushilevitz and Nisan [KN97] and a more recent one by Rao and Yehudayoff [RY20].

[BBM12] achieved an $\Omega(k)$ lower bound for (adaptive) junta testing, matching the previous best result of [CG04], by providing an $\Omega(k)$ lower bound for the problem of k-linearity testing. Recall that linear Boolean functions are just functions of the form $f(x) = \bigoplus_{i \in S} x_i =: \chi_S(x)$ for some $S \subseteq [n]$. Their argument is so simple that we can sketch it here briefly. It is based on a reduction from the *disjointness* problem in communication complexity, in which Alice has input x and Bob has input y, and both x and y have hamming weight k (k nonzeros), and they are trying to figure out if x and y are disjoint by exchanging as little information as possible. Alice can thus form $f = \chi_A$ (where A is the set of nonzero coordinates of her input), and likewise Bob can form $g = \chi_B$. It is not hard to see that $h = f \oplus g$ is $2k$-linear if A and B are disjoint, otherwise it is at most $2k - 2$-linear. Note that it is also not hard to argue that any $2k - 2$-linear function is at least $1/2$-far from being $2k$-linear. Therefore, Alice can run a $2k$-linearity tester on h. In more detail, whenever the testing algorithm tries to query h on z, Alice asks Bob to send over $\chi_B(z)$, and then uses

$\chi_A(z) \oplus \chi_B(z)$ as the answer to the testing algorithm's query. The total number of bits communicated is thus the total number of queries made by the tester for h. It is well known that the communication complexity of the disjointness problem is $\Omega(k)$, which immediately implies the $\Omega(k)$ linearity testing lower bound, as claimed.

While the lower bound of [BBM12] improved what was known for linearity testing, and exhibited an exciting and simple new connection between property testing and communication complexity, it did not improve the $\Omega(k)$ lower bound found in [CG04]. It did, however, pave the way for the final improvement, which came in a recent work in 2018 of Sağlam [Sag18]. The lower bound in [Sag18] was also proven via the connection between communication complexity and property testing. In particular, the communication complexity problem considered in [Sag18] was the *k-hamming distance problem*, wherein Alice and Bob are given two arbitrary subsets A, B of $[n]$ and must determine if the intersection is at most k. By itself, this problem does not reduce to k-linearity testing as nicely as the k-disjointness problem does, but luckily [Sag18] proved a lower bound for the even simpler problem, wherein Alice and Bob are allowed to answer arbitrarily if $\|x-y\|_1 \notin \{k-2, k, k+2\}$. In particular, [Sag18] showed the following lemma.

Lemma 1.6.1 ([Sag18], Theorem 1.8)**.** *For $k^2 \leq \delta n$, the randomized communication complexity of k-Hamming distance problem is $\Omega(k \log(k/\delta))$. Notably, the bound applies even to protocols that are allowed to answer arbitrarily/incorrectly when $\|x-y\|_1 \notin \{k-2, k, k+2\}$.*

The caveat that Alice and Bob can answer arbitrarily for most inputs is actually crucial to maintain the reduction we saw in [BBM12], which we leave as an exercise for the reader. The proof of Lemma 1.6.1, unlike the simple arguments of [CG04] and [BBM12], is actually quite involved, utilizing information theoretic techniques and resolving longstanding conjectures of Blakley, Dixon, Erdös, and Simonovits. Interestingly, one cannot prove such a strong lower bound if one is only required to answer correctly when $\|x-y\|_1 \in \{k, k+2\}$. Interesting, the proof actually resembles the lower bound proof of [FKR$^+$04] in that it analyzes a random walk, and looks at the statistical distance between steps that are two timesteps apart. With Lemma 1.6.1 in hand and the reduction technique of [BBM12], we have finally closed the gap between the upper and lower bounds for adaptive junta testing (for constant ε, δ), which by the algorithm of [Bla09] is $\Theta(k \log k)$.

1.6.2 Nonadaptive Lower Bounds

In the previous subsection, we have extensively discussed adaptive lower bounds for the k-junta problem. In Section 1.5, we saw that the best adaptive algorithm ([Bla09]) had query complexity $O(k \log k + k/\varepsilon)$, while the best nonadaptive algorithm ([Bla08]), had query complexity $\widetilde{O}(k^{3/2})$ (see Table 1.1). The natural question, which was open for many years, was whether this gap was inherent or not. Equivalently, one may wonder whether adaptivity truly helps for junta testing, or if there was a better nonadaptive algorithm out there somewhere, with query complexity $\widetilde{O}(k)$. It turns out, for junta testing, adaptivity strictly helps us. However, this was not immediately clear. Buhrman et. al. [BGMdW13] first gave an $\Omega(k \log k)$ lower bound (for constant ε) for nonadaptively testing k-linearity, which extends trivially to an $\Omega(k \log k)$ lower bound for testing k-juntas nonadaptively. This, however, matched the upper bound of [Bla09], so it was still unclear whether adaptivity helped. This changed in 2015 when Servedio, Tan, and Wright showed definitively that adaptivity helps. In particular, they showed that for all ε such that $k^{-o_k(1)} \leq \varepsilon \leq o_k(1)$, any nonadaptive k-junta tester must make

$$\Omega\Big(\frac{k\log k}{\varepsilon^c \log(\log(k)/\varepsilon^c)}\Big)$$

queries, where $c < 1$ is an absolute constant. This expression is somewhat cumbersome to parse, but the main takeaway is that for certain restricted values of ε such as $\varepsilon = 1/\log k$, this lower bound is asymptotically larger than the $O(k\log k + k/\varepsilon)$ upper bound given in [Bla09].

The scene changed more dramatically in 2017, when Chen et. al. [CST+18] exhibited a $\widetilde{\Omega}(k^{3/2}/\varepsilon)$ query lower bound for nonadaptive junta testing. This more dramatically affirmed that adaptivity is helpful for junta testing and also confirmed that the $\widetilde{O}(k^{3/2})/\varepsilon$ algorithm of [Bla08] is optimal up to polylogarithmic factors. [CST+18] proved their lower bound via Yao's principle, by defining distributions that (with high probability) were either k-juntas or ε-far from k-juntas. Their choice of distribution allowed them to reduce to a simpler problem of trying to determine the size of an unknown set (of size roughly k), with queries than only give partial information about the set. For this simpler problem, they gave an $\widetilde{\Omega}(k^{3/2}/\varepsilon)$ lower bound, and we refer the interested reader to their paper for more details.

In short, we have seen that the adaptive query complexity of junta testing has been settled up to constants and the dependence on ε, while the nonadaptive query complexity of testing juntas has been settled up to polylogarithmic factors.

1.6.3 Quantum Lower Bounds

We have already mentioned a couple of the results in quantum lower bounds in Section 1.5, but we repeat them here in a bit more detail for completeness. The original work of Atici and Servedio [AS07] gave an $\Omega(\sqrt{k})$, but this came with the caveat that it only applied to algorithms that were "Fourier sampling based". In more detail, [BJ99] showed that in the quantum oracle model, it is possible to simulate a draw of a set $S \subseteq [n]$ according to the Fourier spectrum of f (i.e. S is sampled with probability $\widehat{f}(S)^2$) with probability $1-\delta$ using only $O(\log(1/\delta))$ uniform quantum examples.[13] For the reader unfamiliar with Boolean Fourier/harmonic analysis, we refer to the excellent book by O'Donnell as well as Section 1.9. The reason that such Fourier samples are interesting is precisely because quantum algorithms can do them so easily, while it seems to be very difficult for classical algorithms.[14] As a demonstration to why these Fourier samples might be helpful for the problem of k-junta testing, [AS07] point out that using Fourier samples to f, one can easily distinguish between the two "hard" distributions that [CG04] used in their $\Omega(k)$ lower bound. Thus, [AS07] had define slightly more nuanced distributions to apply Yao's principle on, which resulted in their $\Omega(\sqrt{k})$ lower bound (assuming the only additional quantum power the algorithm used was Fourier sampling).

More generally, [ABRdW16] point out that distinguishing between functions that depend only on $k+O(1)$ variables and functions that depend on k variables is easy, since Lemma 2.14

[13]Uniform quantum examples simply take in the state $|1^n, 1\rangle$ and map it to a uniform superposition over all queries and their answers, $\sum_x \frac{1}{2^{n/2}}|x, f(x)\rangle$. We note that Fourier Sampling can be achieved with only one uniformly random sample if one simply applies a Hadamard unitary operator after the uniform query.

[14]Interestingly, as we discuss later in the result of [ITW21] which we spend Chapter 2 discussing can be seen as attempting to sample from the Fourier distribution (for interesting coordinates) above level $\sqrt{k}$. Even more interestingly, we do not see immediately how quantum computers and exact Fourier sampling may help reduce the query complexity beyond the current $2^{\widetilde{O}(\sqrt{k})}$ barrier for the problem of tolerant junta testing.

in [ABRdW16] shows that at least one of the $O(1)$ "extra" variables must have influence at least $\Omega(1)$, and therefore will be detected with very high probability with only $\log k$ quantum Fourier sampling queries. Hence, the distributions found in [AS07] are distributions that are either over k inputs or $k + \Omega(k)$ variables. However, the techniques only give a lower bound for nonadaptive quantum testers, and the $\widetilde{O}(\sqrt{k})$ quantum algorithm given in [ABRdW16] used adaptive queries. Towards proving tight upper and lower bounds for the adaptive query complexity of quantum junta testing, [ABRdW16] gave an $\Omega(k^{1/3})$ adaptive query lower bound via a reduction from the *collision* problem, in which, at a high level, one must determine whether a function is 2-to-1 or 1-to-1. An $\Omega(k^{1/3})$ quantum query lower bound (and matching upper bound) was already known for this problem by previous work, so an $\Omega(k^{1/3})$ query lower bound for quantum junta testing (via the reduction of [ABRdW16]) was directly implied. It was conjectured in [ABRdW16] that the true adaptive lower bound for quantum junta testing should be $\Omega(\sqrt{k/\varepsilon})$, but nothing better is currently known.

In Table 1.2, we summarize the known lower bound results and their corresponding upper bounds.

Reference	Model	Adaptive?	Query Lower Bound	Corresponding Upper bound
[CST+18]	Classical	No	$\widetilde{\Omega}(k^{3/2}/\varepsilon)$	$\widetilde{O}(k^{3/2})/\varepsilon$ [Bla08]
[CG04, BBM12]	Classical	Yes	$\Omega(k)$	$O(k\log k + k/\varepsilon)$ [Bla09]
[Sag18]	Classical	Yes	$\Omega(k\log k)$	$O(k\log k + k/\varepsilon)$ [Bla09]
[AS07]	Quantum	No	$\Omega(\sqrt{k})$	$O(k/\varepsilon)$ [AS07]
[ABRdW16]	Quantum	Yes	$\Omega(k^{1/3})$	$\widetilde{O}(\sqrt{k/\varepsilon})$ [ABRdW16]

Table 1.2: Nontolerant Junta Testing Lower Bounds

1.7 Tolerant Junta Testing

Now that we have thoroughly explored the work that has been done in the context of standard junta testing, we are ready to begin our discussion of the current state of tolerant junta testers.

We start by making the following (parameterized) definition of a tolerant tester. In the following we denote by $\mathcal{J}_{n,k}$ the class of k-juntas on n variable Boolean functions, and for a class of functions $\mathcal{C}$.

Definition 1.7.1. *For constants* $0 < c_\ell < c_u < 1/2$ *and a given* $k', k \in \mathbb{N}$ *with* $k' \geq k$, *a* (k, k', c_ℓ, c_u) tolerant junta tester *is an algorithm that, given oracle access to* $f : \{\pm 1\}^n \to \{\pm 1\}$,

1. *if* $\mathsf{dist}(f, \mathcal{J}_{n,k}) \leq c_\ell$ *accepts with probability* $2/3$;
2. *if* $\mathsf{dist}(f, \mathcal{J}_{n,k'}) \geq c_u$ *rejects with probability* $2/3$.

There are already a lot of parameters in this definition, so it is important to parse it carefully. In words, a (k, k', c_ℓ, c_u) distinguishes (with constant probability) between functions that are c_ℓ-close to k juntas, and c_u-far from all k'-juntas.

Our definition is convenient because it incorporates both tolerant and parameterized testers. For example, when $c_\ell = 0$ the tester is non-tolerant and when $k' = k$ the tester is

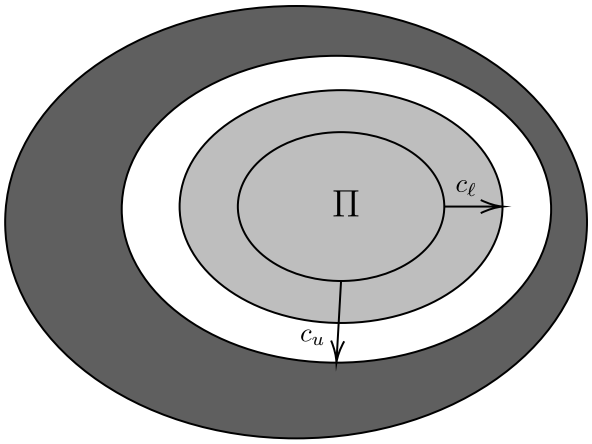

Figure 1.1: A visualization of the tolerant property testing paradigm. Assuming the outermost oval represents all functions $f : \{\pm 1\}^n \to \{\pm 1\}$ and the property at hand is represented by a class of functions Π, the goal is to distinguish between the light grey (at most c_ℓ close to a function in Π) and the dark grey (at least c_u far from all functions in Π) regions.

non-parameterized.[15] Parnas, Ron, and Rubinfeld in their seminal work [PRR04] showed that while standard property testers, when querying uniformly, are weakly tolerant, entirely new algorithms are usually needed to tolerant test with better parameters. Tolerant junta testing was first considered by Diakonikolas et al. [DLM+07] which used the aforementioned observation from [PRR04] to show that a standard tester from [FKR+04] actually gave a $(k, k, \mathsf{poly}(\frac{\gamma}{k}), \gamma)$ tolerant tester. Chakraborty et al. [CFGM12] subsequently showed that a similar analysis to that of Blais [Bla09] gave a $(k, k, \gamma/C, \gamma)$ tolerant junta tester (for some constant C) using $\exp(k/\gamma)$ queries.

More recently, Blais et al. [BCE+19, Theorem 1.2] showed a concrete tradeoff between the query complexity and the amount of tolerance. In particular, they gave an algorithm which, given k, γ, and $\rho \in (0, 1)$, is a $(k, k, \rho\gamma/16, \gamma)$ tolerant junta tester. The query complexity of the algorithm is $O\left(\frac{k \log k}{\gamma\rho(1-\rho)^k}\right)$. This expression requires parsing; note that when ρ is a constant bounded away from zero, this yields an $\exp(k)$ query algorithm, but when $\rho = 1/k$ this yields a $\mathsf{poly}(k)$ query algorithm. We also note that there is an undesirable multiplicative "gap" between c_u and c_ℓ that precludes one from tolerantly testing for arbitrary close values of c_u and c_ℓ (i.e., in [BCE+19], $c_u \geq 16c_\ell$ for all choices of ρ). For example, one could not distinguish between functions that are 0.05-close and 0.1-far from being a k-junta with this algorithm. The recent work of De, Mossel, and Neeman [DMN19] addressed this, giving an algorithm for any arbitrary $\gamma, \varepsilon > 0$ that required $2^k \mathsf{poly}(k, \frac{1}{\varepsilon})$ queries and was a $(k, k, \gamma, \gamma + \varepsilon)$ tolerant junta tester.

In the relaxed setting (when $k' \neq k$), [BCE+19, Theorem 1.1] also gave an algorithm which used $\mathsf{poly}(k, \frac{1}{\gamma})$ queries to f and was a $(k, 4k, \gamma/16, \gamma)$ tolerant junta tester. While k' has a good dependence on k in this theorem, this once again posed the issue of not allowing for arbitrary c_u and c_ℓ values. This was again resolved by [DMN19, Corollary 1.6], which

[15] We note that in the above definition we upper bound $c_u < 1/2$ since k-juntas are closed under complements, meaning if $g \in \mathcal{J}_{n,k}$, then $-g \in \mathcal{J}_{n,k}$.

gave a $(k, O(k^2/\varepsilon^2), \gamma, \gamma+\varepsilon)$ tolerant junta tester with query complexity $\mathsf{poly}(k, \frac{1}{\varepsilon})$ at the expense of k' having a worse dependence on k.

It is interesting to note that the techniques used to obtain the results from [BCE+19] and [DMN19] are quite different, and yield results that are qualitatively similar but quantitatively incomparable. The results from [BCE+19] extend the techniques of [FKR+04, Bla08, Bla09], which partition the n input coordinates into $\mathsf{poly}(k)$ disjoint sets or "buckets". As discussed in Section 1.5, it is immediate that any k-junta is a k-part junta, but in [Bla09] it was shown that with high probability a function that is far from being a k-junta is also far from being a "k-part junta". The results of [BCE+19] extend the idea of considering the relationship between k-juntas and k-part juntas in the context of tolerant testing. In particular, they showed the following (easy to prove) lemma.

Lemma 1.7.2 (Lemma 3.3 in [BCE+19])**.** *For $f : \{\pm1\}^n \to \{\pm1\}$ and $k \geq 1$, let $\alpha = \mathsf{dist}(f, \mathcal{J}_{n,k})$ and let $\mathcal{I}$ be any partition of $[n]$ into $\ell \geq k$ parts. Then there exists a collection of $r \leq k$ of the buckets $I_{i_1}, ..., I_{i_r} \in \mathcal{I}$ such that, defining $V = \bigcup_{j=i_1}^{i_r} I_j$,*

$$\mathbf{Inf}_V[f] \leq 4\alpha.$$

In other words, f is 2α close to a k-part junta with respect to the partition $\mathcal{I}$.

Lemma 1.7.2 combined with Lemma 1.5.2, which we already saw in Section 1.5 in the context of nontolerant testers, give a natural approach towards tolerant junta testing, which is exactly the approach taken in [BCE+19]. The idea is to partition f's inputs into $O(k^2)$ buckets $\mathcal{I}$ and then to determine how close f is to a k-part junta with respect to the partition $\mathcal{I}$. A brute force algorithm doing this would take approximately $\binom{O(k^2)}{k} = 2^{\widetilde{O}(k)}$ queries. [BCE+19] were able to achieve their main result of $O\left(\frac{k \log k}{\gamma\rho(1-\rho)^k}\right)$ queries by taking advantage of the fact that **Inf** is a *submodular* function, and thus utilize state of the art algorithms for submodular function optimization under cardinality constraints[16] as well as a clever technique to recycle queries.

On the other hand the techniques in [DMN19] suggest a completely new way of attacking the problem of tolerant k-junta testing, and indeed junta testing in general. The core idea in [DMN19] was to get access to "oracles" to coordinates of f which have large low-degree influence. Here, when we say coordinate oracle for input i, we mean we have constructed/found a (randomized) function g such that, with high probability, $g(x) = \pm x_i$. These *coordinate oracles* are obtained with high probability via a combination of random restrictions and noise operators to the original function. In Section B we extensively discuss how [DMN19] achieved access to these oracles for influential coordinates. Intuitively, it suffices to only consider coordinates/inputs to f which have a large influence up to a small loss in correlation (this is formalized in Claim 2.2.6). Once we have obtained access to these coordinates with large influence, they can be used to search, in a brute force manner, for the nearest k-junta, resulting in the exponential query algorithm of [DMN19].

1.7.1 Tolerant Junta Testing Lower Bounds

In terms of lower bounds for tolerant testing of juntas, two recent works addressed the non-adaptive case. Levi and Waingarten [LW19] demonstrated that there exists $0 < \varepsilon_1 < \varepsilon_2 < 1/2$ such that any $(k, k, \varepsilon_1, \varepsilon_2)$ tolerant junta tester requires $\widetilde{\Omega}(k^2)$ non-adaptive queries to

[16]One must optimize over subsets of a certain size for this problem.

f. In particular, this result demonstrated that the tolerant testing regime is quantitatively harder than the standard testing regime, in which the $\widetilde{O}(k^{3/2})$-query non-adaptive query algorithm of [Bla08] is known (and indeed optimal due to [CST$^+$18]). Subsequently, Pallavoor, Raskhodnikova, and Waingarten [PRW19] demonstrated that for any $k \leq n/2$ there exists $0 < \varepsilon_1 < \varepsilon_2 < 1/2$ (with $\varepsilon_1 = O(1/k^{1-\eta})$ and $\varepsilon_2 = \Omega(1/\sqrt{k})$) such that every nonadaptive $(k, k, \varepsilon_1, \varepsilon_2)$-tolerant junta tester requires at least 2^{k^η} queries to f, for any $0 < \eta < 1/2$. [17] This exponentially improved the best known lower bounds, and indeed showed the tolerant testing problem, at least in the case of nonadaptive algorithms, is *exponentially* harder than the nontolerant testing of juntas.

We discuss the techniques of [PRW19] here briefly, since they are novel, interesting, and not terribly difficult to state. One of the major techniques utilized in [PRW19] to great effect is the notion of an *erasure resilient tester*. In this model, some of f's outputs have been erased, so that now $f : \{0,1\}^n \to \{0, 1, \perp\}$, where $\perp$ represents an erased value.

Definition 1.7.3 (Erasure Resilient Testing)**.** *We say that an algorithm $\mathcal{A}$, given query access to $f : \{0,1\}^n \to \{0, 1, \perp\}$, is an α-erasure resilient ε-tester for property Π if for any f with at most α fraction of its outputs erased, $\mathcal{A}$ accepts with probability $2/3$ if the erasures can be filled in so that $f \in \Pi$, and $\mathcal{A}$ rejects with probability $2/3$ if for* every *completion of erasures results in a function that is ε-far from Π.*

What is nice about this model is that its query complexity lies somewhere in between the standard property testing model and the tolerant testing model. More specifically, a standard tester is simply a special case of an erasure resilient tester with α set to zero. Moreover, and crucially for [PRW19], a tolerant tester that accepts functions α-close to Π and rejects functions that are ε-far (for $\varepsilon > \alpha$), can be used to get an α-erasure resilient ε-tester. The reason is simple: the erasure resilient tester can just fill in the erasures with random values and run the tolerant tester. [PRW19] proved their lower bound for erasure resilient testers, which naturally extended to a lower bound for tolerant testing of juntas.[18]

As is common, [PRW19] uses Yao's principle for their lower bound. In order to do so, they define two distributions, $\mathcal{D}^+$ and $\mathcal{D}^-$, which are designed such that there exists a completion of the erasures that makes $\mathbf{f}^+ \sim \mathcal{D}^+$ a $n/2$-junta which has at most α fraction of its outputs erased, and $\mathcal{D}^-$, which is designed so that no completion of the erasures of $\mathbf{f}^- \sim \mathcal{D}^-$ results in a function that is ε-close to being a $n/2$-junta. The distribution of $\mathcal{D}^-$ is visualized in Figure 1.2. The procedure for sampling $\mathbf{f}^- \sim \mathcal{D}^-$ and $\mathbf{f}^+ \sim \mathcal{D}^+$ is as follows, assuming n is a multiple of 4. Let $\eta \in (0, 1/2)$.

1. Sample a uniformly random subset $[n]$ of size $n/2$, and call it M. If $|x_M| > n/4$, set $\mathbf{f}^-(x) = 1$, and otherwise if $|x_M| < n/4$, set $\mathbf{f}^-(x) = 0$.

2. For each x with $|x_M| = n/4$ (so the exact middle layer of the subcube $\{0,1\}^M$ (note this has size $\Theta(2^{n/2}/\sqrt{n})$ via Stirling's formula) if $|x_{\overline{M}}| \in [n/4 - n^\eta, n/4 + n^\eta]$, set $f(x) = \perp$. Otherwise we do the following:

   (a) For $\mathcal{D}^-$, $f(x) = b_{x_M}$ if $|x_{\overline{M}}| > n/4 + n^\eta$ and $f(x) = 1 - b_{x_M}$ if $|x_{\overline{M}}| < n/4 + n^\eta$, for a random bit b_{x_M}.

[17] We note that this lower bound does not necessarily rule out $\mathsf{poly}(k)\exp(1/\varepsilon)$ nonadaptive query $(k, k, \varepsilon_1, \varepsilon_2)$ (where $\varepsilon = \varepsilon_2 - \varepsilon_1$) tolerant junta testers due to the setting of ε_1 and ε_2 in their hard instance.

[18] The techniques in [PRW19] also give lower bounds for the tolerant testing of monotonicity and unateness, but we just focus on the application to tolerant junta testing.

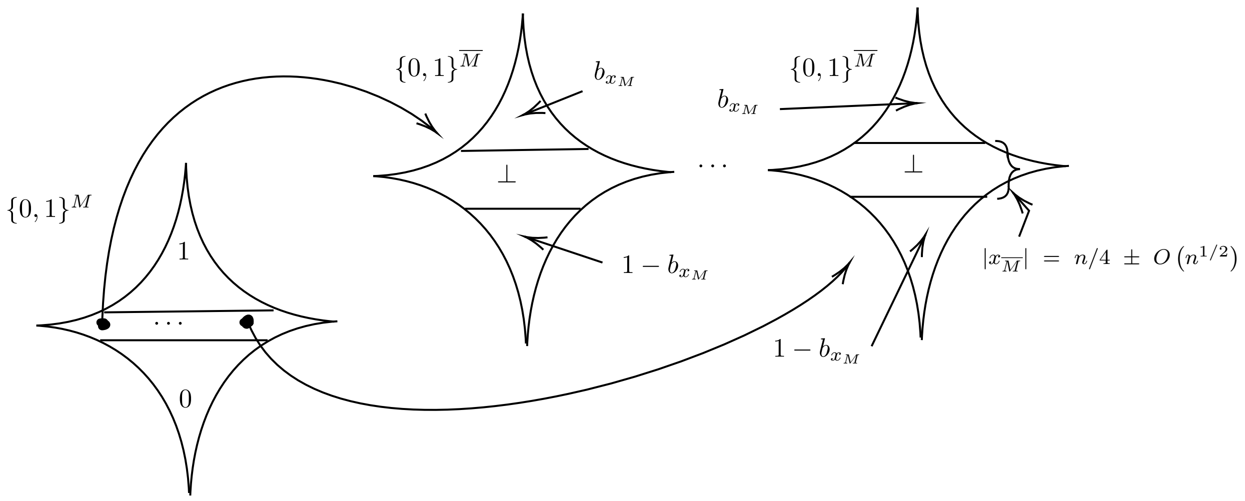

Figure 1.2: Functions sampled according to $\mathcal{D}^-$ in [PRW19], which is designed to be far from all being a k-junta for all completions of the erasures.

(b) For $\mathcal{D}^+$, we make the $\overline{M}$ subcubes "constant" by setting $f(x) = b_{x_M}$ if $|x_{\overline{M}}| > n/4 + n^\eta$ or $|x_{\overline{M}}| < n/4 + n^\eta$, for a random bit b_{x_M}.

Crucially, with very high probability, [PRW19] showed that functions sampled according to $\mathcal{D}^-$ is at least $\varepsilon = \Omega(1/\sqrt{n})$ far from being a $n/2$-junta (for any completion of the erasures), while every function can trivially be completed to be a $n/2$ junta. Note that α is also on the order of $n^\eta/\sqrt{n}$, via Stirling's approximation. Now, we come to the crux of the argument, which is surprisingly simple.

The key is to notice that the only chance any tester has of distinguishing between $\mathbf{f}^- \sim \mathcal{D}^-$ and $\mathbf{f}^+ \sim \mathcal{D}^+$ is to sample two queries x and y such that $x_M = y_M$, yet x and y fall on opposite ends of the $\overline{M}$ subcube, i.e. either

1. $|x_{\overline{M}}| > n/4 + n^\eta$ and $|y_{\overline{M}}| < n/4 - n^\eta$ or
2. $|x_{\overline{M}}| < n/4 - n^\eta$ and $|y_{\overline{M}}| > n/4 + n^\eta$.

Call the event that a nonadaptive tester queries such an (x, y) pair B. Then we have the following lemma:

Lemma 1.7.4 (Claim 4.11 in [PRW19]). *For any nonadaptive algorithm making $q \leq 2^{n^\eta}$ queries, we have*

$$\mathbf{Pr}[B] < 1/8.$$

Thus, with probability at least 7/8 the $\mathbf{f}^- \sim \mathcal{D}^-$ and $\mathbf{f}^+ \sim \mathcal{D}^+$ look exactly the same to the algorithm.

Proof. Note that in order for B to happen, x and y must equal on M but differ in at least $t = 2n^\eta + 2$ coordinates in $\overline{M}$. For a uniformly random M, we have that this happens with probability at most

$$\frac{\binom{n-t}{n/2}}{\binom{n}{n/2}} \le 2^{-t}.$$

A simple union bound over the at most q^2 pairs of points the tester queries completes the argument. □

This concludes the sketch of the lower bound technique in [PRW19]. We now turn to discuss our new upper bounds, originally appearing in [ITW21], before summarizing the upper and lower bounds for tolerant junta testing in Table 1.3.

1.8 New Upper Bounds for Tolerant k-junta Testing

As we have previously noted, the main results of this section and thesis come from [ITW21]. Our first result is a subexponential-in-k query tolerant junta tester in the standard (non-relaxed) setting. In fact, we obtain an ε-accurate estimate of the distance of f to the class of k-juntas.

Theorem 1.8.1. *Given a Boolean function $f : \{\pm 1\}^n \to \{\pm 1\}$, it is possible to estimate the distance of f from the class of k-juntas to within additive error ε with probability $2/3$ using $2^{\widetilde{O}(\sqrt{k/\varepsilon})}$ adaptive queries to f. In particular, when ε is constant, this yields a $2^{\widetilde{O}(\sqrt{k})}$-query algorithm. However, the algorithm still requires $\exp(k/\varepsilon)$ time.*

A simple corollary of the above theorem is that for any $0 < c_\ell < c_u < 1/2$, we have a (c_u, c_ℓ, k, k) tolerant junta tester with the same query complexity as in Theorem 1.8.1, where $\varepsilon = (c_u - c_\ell)/2$. This is an improvement of the results of [DMN19, BCE+19], whose tolerant junta testers when $k' = k$ required exponential query complexity in k in the worst case. We note that although we obtain this improvement, our algorithm still requires $\exp(k)$ time. In Appendix C, we show a result solving a similar problem[19] with an improved dependence on ε, giving an algorithm requiring only $2^{\widetilde{O}(\sqrt{k}\log(1/\varepsilon))}$-queries and $\exp(k\log(1/\varepsilon))$ time (see Theorem 2.3.10).

In the relaxed/parameterized setting when $k' \neq k$, we give a polynomial-in-k query tolerant junta tester that is valid for any setting of c_u and c_ℓ, and reduces k' dependence on k to be linear instead of quadratic due to the result of [DMN19, Corollary 1.6].

Theorem 1.8.2. *For any $\gamma, \varepsilon > 0$ and $k \in \mathbb{N}$, there is an algorithm with query complexity $\mathsf{poly}(k, 1/\varepsilon)$ that is a $(k, O(k/\varepsilon^2), \gamma, \gamma + \varepsilon)$-tolerant junta tester.*

Theorem 1.8.2 is a simple corollary of the following theorem we prove.

Theorem 1.8.3. *Let $\varepsilon > 0$, $k \in \mathbb{N}$, and $k' = O(k/\varepsilon^2)$. Then, there exists an algorithm that given parameters k, ε and oracle access to f makes at most poly$(k, 1/\varepsilon)$ queries to f and returns a number α such that with high probability (at least 0.99)*

1. $\alpha \le \mathsf{dist}(f, \mathcal{J}_{n,k}) + \varepsilon$
2. $\alpha \ge \mathsf{dist}(f, \mathcal{J}_{n,k'}) - \varepsilon$

[19] In particular, this problem is the problem of finding the subset of k inputs that "contain" the most Fourier mass – see Section 1.9 and Theorem 2.3.10 for more details.

Indeed, to solve the problem in Theorem 1.8.2 we can apply the algorithm from Theorem 1.8.3 with $\varepsilon = (c_u - c_\ell)/3$ and accept if and only if $\alpha < \frac{1}{2}(c_u + c_\ell)$. If $\mathsf{dist}(f, \mathcal{J}_{n,k}) \leq c_\ell$ we have that with high probability $\alpha \leq c_\ell + \varepsilon < \frac{1}{2}(c_u + c_\ell)$ and we will accept. On the other hand, if $\mathsf{dist}(f, \mathcal{J}_{n,k'}) \geq c_u$ we have that with high probability $\alpha \geq c_u - \varepsilon > \frac{1}{2}(c_u + c_\ell)$ and we will reject.

Both of the algorithms used to prove Theorem 1.8.1 and Theorem 1.8.3 rely on the fact that we can get approximate oracle access to influential coordinates of f using techniques from [DMN19]. From there, we analyze the Fourier coefficients of f after a series of random restrictions in order gain more information about the relevant coordinates of f at different Fourier levels. Along the way, we give an algorithm which provides us with oracle access to a junta in the following sense:

Theorem 1.8.4 (Informal)**.** *Let $f : \{\pm 1\}^n \to \{\pm 1\}$, $\mathcal{D} = \{g_1, \ldots, g_{k'}\}$ be a set of functions giving oracle access to a certain set of coordinates. Let g be a function from $\{\pm 1\}^{k'} \to [-1, 1]$ defined by $g(x) = \mathbf{E}[f(y) | g_1(y) = x_1, \ldots, g_{k'}(y) = x_{k'}]$. Then g can be computed by a randomized algorithm that runs in expected time $\mathsf{poly}(k')$.*

We note that one can view this as an oracle access to the junta, without even figuring out the coordinates on which the junta depends. More details on the ideas behind both algorithms can be found in Section 2.1.

Now that we have surveyed previous results and stated our main theorems, we can summarize the state of affairs for tolerant junta testing in Table 1.3.

Adaptive?	k'	Lower Bound	Upper bound
No	k	$\Omega(2^{k^{0.499}})$ [PRW19]	None
Yes	k	$\Omega(k \log k)$ [Sag18]	$2^{\widetilde{O}(\sqrt{k/\varepsilon})}$ [ITW21]
Yes	$4k$	None	$\mathsf{poly}(k/\varepsilon)$ [BCE+19]
Yes	$O(k/\varepsilon^2)$	None	$\mathsf{poly}(k/\varepsilon)$ [ITW21]

Table 1.3: Upper and lower bounds in tolerant junta testing. Note that the results of [BCE+19] are only valid for certain settings of c_ℓ and c_u.

1.9 Further Preliminaries

Before we dive into the technical details presented in Chapter 2, we need to establish some more preliminaries.

1.9.1 Probability

We recall the following Chernoff/Hoeffding bounds.

Fact 1.9.1. *If $X_1, \ldots, X_N$ are independent random variables bounded in $[0, 1]$ and $\bar{X} := \frac{1}{N}\sum_{i=1}^{N} X_i$, then we have*

$$\mathbf{Pr}[|\bar{X} - \mathbf{E}[\bar{X}]| \geq \eta] \leq 2\exp(-2N\eta^2),$$

Furthermore, denoting by $p = \mathbf{E}[\bar{X}]$, we have

$$\mathbf{Pr}[\bar{X} \le p - \eta] \le \exp(-2N\eta^2),$$

$$\mathbf{Pr}[\bar{X} \le (1-\eta)p] \le \left(\frac{e^{-\eta}}{(1-\eta)^{1-\eta}}\right)^{pN} \le \exp\left(-\frac{\eta^2 pN}{2}\right).$$

1.9.2 Further Fourier Analytic Tools/Facts

We start with the following claim, which tells us exactly what the nearest junta to f is on a given subset S.

Claim 1.9.2 (Nearest k-junta on a Subset)**.** *For a function $f : \{\pm 1\}^n \to [-1, 1]$ and a subset $T \subseteq [n]$, the Boolean-valued junta-on-T most correlated with f is given by*

$$\mathsf{sgn}(f_{\mathsf{avg},T}(x)) = \mathsf{sgn}\left(\mathop{\mathbf{E}}_{y\in\{\pm1\}^n}[f(y)|y_T = x_T]\right).$$

Furthermore, the correlation between f and $\mathsf{sgn}(f_{\mathsf{avg},T}(x))$ is simply $\mathbf{E}_{x\sim\{\pm1\}^n}[|f_{\mathsf{avg},T}(x)|]$.

Proof. Let $g : \{\pm1\}^n \to [-1, 1]$ be any junta-on-T. It suffices to show that $\mathbf{E}_x[f(x)g(x)] \le \mathbf{E}[f(x)\mathsf{sgn}(f_{\mathsf{avg},T}(x))]$, as we do next. Indeed, for any $g(x)$ that is a junta-on-T we have $g(x) = g'(x_T)$ for some $g' : \{\pm1\}^T \to [-1, 1]$. Thus, we have

$$\begin{aligned}
\mathop{\mathbf{E}}_{x\sim\{\pm1\}^n}[f(x)g(x)] &= \mathop{\mathbf{E}}_{x\sim\{\pm1\}^n}[f(x)g'(x_T)] \\
&= \mathop{\mathbf{E}}_{x\sim\{\pm1\}^n}\left[g'(x_T)\cdot \mathop{\mathbf{E}}_{y\sim\{\pm1\}^n}[f(y)|x_T = y_T]\right] \\
&= \mathop{\mathbf{E}}_{x\sim\{\pm1\}^n}[g'(x_T)f_{\mathsf{avg},T}(x)] \\
&\le \mathop{\mathbf{E}}_{x\sim\{\pm1\}^n}[|f_{\mathsf{avg},T}(x)|] \\
&= \mathop{\mathbf{E}}_{x\sim\{\pm1\}^n}[\mathsf{sgn}(f_{\mathsf{avg},T}(x))\cdot f_{\mathsf{avg},T}(x)] \\
&= \mathop{\mathbf{E}}_{x\sim\{\pm1\}^n}[f(x)\mathsf{sgn}(f_{\mathsf{avg},T}(x))].
\end{aligned}$$ □

We will also be considering the Fourier mass at different levels of the spectrum, as defined below.

Definition 1.9.3. *For a function $f : \{\pm1\}^n \to \mathbb{R}$ we define:*

$$\mathbf{W}^{\le k}[f] = \sum_{|S|\le k} \widehat{f}(S)^2 .$$

The definitions of $\mathbf{W}^{\ge k}[f]$, $\mathbf{W}^{=k}[f]$, and similar follow from a natural extension.

Another useful tool in Boolean Function Analysis is the noise operator T_ρ, which intuitively collapses the Fourier mass of f to the lower levels. For a vector $x \in \{\pm1\}^n$ we denote by $N_\rho(x)$ the distribution over vectors $y \in \{\pm1\}^n$ such that for each coordinate $i \in [n]$ independently $y_i = x_i$ with probability $(1+\rho)/2$ and $y_i = -x_i$ otherwise (alternatively,

$\mathbf{E}[x_i y_i] = \rho$). For a function $f : \{\pm 1\}^n \to \mathbb{R}$ we denote by $T_\rho f : \{\pm 1\}^n \to \mathbb{R}$ the function defined by

$$T_\rho f(x) = \mathop{\mathbf{E}}_{y \sim N_\rho(x)}[f(y)]$$

There's also a nice Fourier expression for the function $T_\rho f$ given by $T_\rho f(x) = \sum_{S \subseteq [n]} \widehat{f}(S)\rho^{|S|}$. We will need a simple fact about the noise operator.

Fact 1.9.4. *For any function $f : \{\pm 1\}^n \to \mathbb{R}$ and any $\rho \in [-1, 1]$ we have that $\mathbf{E}[|T_\rho f|] \leq \mathbf{E}[|f|]$.*

Proof.

$$\mathop{\mathbf{E}}_{x \sim \{\pm 1\}^n}[|T_\rho f(x)|] = \mathop{\mathbf{E}}_{x \sim \{\pm 1\}^n}[|\mathop{\mathbf{E}}_{y \sim N_\rho(x)}[f(y)]|] \leq \mathop{\mathbf{E}}_{x \sim \{\pm 1\}^n}[\mathop{\mathbf{E}}_{y \sim N_\rho(x)}|f(y)|] = \mathop{\mathbf{E}}_{y \sim \{\pm 1\}^n}[|f(y)|]. \quad \square$$

We will also make use of random restrictions, which, similar to the noise operator T_ρ, intuitively push the Fourier mass of f onto the lower levels and often simplify f significantly.

Definition 1.9.5 (Restriction)**.** *Consider the class of functions on $\{\pm 1\}^n$. A restriction is a pair (J, z) where $J \subseteq [n]$, and $z \in \{\pm 1\}^{\overline{J}}$. Given a function $f : \{\pm 1\}^n \to \mathbb{R}$, and a restriction (J, z), the* restricted function *$f_{\overline{T} \to z} : \{\pm 1\}^T \to \mathbb{R}$ is defined by $f_{\overline{T} \to z}(x) = f(y)$ where $y_T = x$ and $y_{\overline{T}} = z$.*

Definition 1.9.6 (δ-Random Restriction)**.** *For $\delta \in [0, 1]$ we say that J is a δ-*random subset *of S if it is formed by including each element independently with probability δ, which we denote as $J \subseteq_\delta S$. A δ-random restriction, denoted $(J, z) \sim \mathcal{R}_\delta$, is sampled by taking J to be a δ-random subset J on $[n]$, and taking z to be a uniformly random string in $\{\pm 1\}^{\overline{J}}$.*

Occasionally, we will abuse notation and think of $f_{\overline{T} \to z}$ as a function from $\{\pm 1\}^n$ to $\{\pm 1\}$ that ignores bits outside T. For example, $f_{\overline{T} \to z} : \{\pm 1\}^n \to \{\pm 1\}$ is given by $f_{\overline{T} \to z}(x) = f(x_T, z_{\overline{T}})$. Finally, we will use the following fact on random restrictions:

Fact 1.9.7. *For a function $f : \{\pm 1\}^n \to \mathbb{R}$ and sets $S \subseteq J \subseteq [n]$ we have*

$$\mathop{\mathbf{E}}_{z \in \{\pm 1\}^{\overline{J}}}[\widehat{f_{\overline{J} \to z}}(S)^2] = \sum_{R \subseteq [n], R \cap J = S} \widehat{f}(R)^2.$$

1.9.3 Estimating Fourier Coefficients

The following claim is a standard tool in many learning algorithms. It establishes that estimating Fourier coefficients of a Boolean function f can be done with a few queries to f.

Claim 1.9.8. *Suppose $f : \{\pm 1\}^n \to \{\pm 1\}$ and $S \subseteq [n]$ then there exists an algorithm that estimates $\widehat{f}(S)$ up to additive error ε with probability at least $1 - \delta$ that makes $O((1/\varepsilon^2) \cdot \log(1/\delta))$ samples.*

Proof Sketch. Estimate $\widehat{f}(S)$ by sampling $m = O((1/\varepsilon^2) \cdot \log(1/\delta))$ uniformly random inputs $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}$ and taking the empirical mean of $\frac{1}{m}\sum_{i=1}^m f(\mathbf{x}^{(i)}) \cdot \chi_S(\mathbf{x}^{(i)})$. The claim follows from Fact 1.9.1. $\square$

The next claim generalizes the claim to a bounded function $f : \{\pm 1\}^n \to [-1, 1]$. For that generalization, we need the definition of a randomized algorithm computing a bounded function f.

Definition 1.9.9 (Randomized Algorithm for a Bounded Function)**.** *Let $f : \{\pm 1\}^n \to [-1, 1]$ be a bounded function. We say that algorithm A is a randomized algorithm for f if on any fixed input x algorithm A outputs a random bit $\mathbf{y} \in \{\pm 1\}$ with $\mathbf{E}[\mathbf{y}] = f(x)$.*

Claim 1.9.10. *Let $f : \{\pm 1\}^n \to [-1, 1]$, and let A be a randomized algorithm for f. Then, there exists an algorithm making $O((1/\varepsilon^2) \cdot \log(1/\delta))$ calls to A that estimates $\widehat{f}(S)$ up to additive error ε with probability at least $1 - \delta$.*

Proof Sketch. We estimate $\widehat{f}(S)$ by sampling $m = O((1/\varepsilon^2) \cdot \log(1/\delta))$ uniformly random inputs $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}$, applying A to each of them to get random bits $(\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_m)$, and taking the empirical mean of $\frac{1}{m} \sum_{i=1}^{m} \mathbf{y}_i \cdot \chi_S(\mathbf{x}^{(i)})$. Note that for each $i \in [m]$ we have that $\mathbf{y}_i \cdot \chi_S(\mathbf{x}^{(i)})$ is a $\{\pm 1\}$ random variable with expectation

$$\mathop{\mathbf{E}}_{\mathbf{x}^{(i)}, \mathbf{y}_i}[\mathbf{y}_i \cdot \chi_S(\mathbf{x}^{(i)})] = \mathop{\mathbf{E}}_{\mathbf{x}^{(i)}}\left[\mathop{\mathbf{E}}_{\mathbf{y}_i}[\mathbf{y}_i | \mathbf{x}^{(i)}] \cdot \chi_S(\mathbf{x}^{(i)})\right] = \mathop{\mathbf{E}}_{\mathbf{x}^{(i)}}[f(\mathbf{x}^{(i)}) \cdot \chi_S(\mathbf{x}^{(i)})] = \widehat{f}(S).$$

The claim follows from Fact 1.9.1. $\square$

Chapter 2

New Algorithms for Tolerant k-junta Testing

2.1 Overview of Techniques

Both of our algorithms rely on only having to consider a subset of influential coordinates, rather than all n input variables. This is obtained using results from [DMN19], and is discussed further in Section 2.2 (and the approach of [DMN19] for obtaining the oracles is discussed in Section 2.2.1). For now, we simply assume that we are only dealing with $\mathsf{poly}(k, 1/\varepsilon)$ coordinates $\mathcal{S}$. For simplicity of presentation, we ignore dependence on ε, and focus only the dependence on k. Thus, in this section, assume that ε is a small universal constant, e.g., $\varepsilon = 0.01$.

2.1.1 Techniques for Establishing Theorem 1.8.3

Our first result shows how to further reduce the number of coordinates we need to consider down to $O(k/\varepsilon^2)$, while only losing at most ε amount of correlation with the maximally correlated k-junta. In establishing Theorem 1.8.3, we first develop intuition behind a new notion of normalized influence that we introduce next:

Definition 2.1.1 (Normalized Influence)**.** *Let $f : \{\pm 1\}^n \to \mathbb{R}$. We define the normalized influence of coordinate i on f as*

$$\mathbf{NInf}_i[f] = \sum_{S \ni i} \frac{\widehat{f}(S)^2}{|S|}.$$

We also naturally define the normalized influence below level k:

$$\mathbf{NInf}_i^{\leq k}[f] := \sum_{\substack{|S| \leq k \\ S \ni i}} \frac{\widehat{f}(S)^2}{|S|}.$$

The next claim states that the sum of normalized influences of f equals its variance.

Claim 2.1.2. *For any function $f : \{\pm 1\}^n \to \mathbb{R}$, we have that $\sum_i \mathbf{NInf}_i[f] = \mathbf{Var}[f]$.*

Proof. We have that

$$\sum_{i\in[n]} \mathbf{NInf}_i[f] = \sum_{i\in[n]} \sum_{S\ni i} \frac{\widehat{f}(S)^2}{|S|} = \sum_{\substack{S\subseteq[n]\\ S\neq\emptyset}} \sum_{i\in S} \frac{\widehat{f}(S)^2}{|S|} = \sum_{\substack{S\subseteq[n]\\ S\neq\emptyset}} |S|\frac{\widehat{f}(S)^2}{|S|} = \sum_{\substack{S\subseteq[n]\\ S\neq\emptyset}} \widehat{f}(S)^2 = \mathbf{Var}[f],$$

where the last equality follows from Parseval's identity. □

Remark 2.1.3. We note that for a balanced Boolean function f (that is, one where $\mathbf{E}_x[f(x)] = 0$) the normalized influences form a probability distribution on the coordinates i.

The idea behind establishing Theorem 1.8.3 begins with the observation the these normalized influences can be thought of as defining a sub-probability distribution over the input coordinates of f, since these are non-negative numbers whose sum is at most 1. The weight assigned to coordinate i, similar to the regular influence, captures how important i is to f, but assigns a higher relative weight to the coordinates with Fourier mass coming from the lower levels of the Fourier decomposition.

The second important observation for us is that for any set T of size at most k we can write

$$\sum_{i\in T} \mathbf{NInf}_i^{\leq k}[f] = \sum_{i\in T} \sum_{\substack{|S|\leq k\\ \emptyset\neq S\ni i}} \frac{\widehat{f}(S)^2}{|S|} \geq \sum_{i\in T} \sum_{\substack{S\subseteq T\\ S\ni i}} \frac{\widehat{f}(S)^2}{|S|} = \sum_{\emptyset\neq S\subseteq T} \widehat{f}(S)^2. \tag{2.1}$$

Intuitively, this shows that if some set of coordinates captures large amount of Fourier mass, then this same subset of coordinates also is very likely to be sampled by our sub-probability distribution defined by the normalized influences. Our idea follows this line of thought – we get decent estimates for all of the normalized influences, and sample coordinates from this estimated distribution. Let T be the "target set" of size k, i.e., the one for which the closest k-junta to f is a junta on T. Without loss of generality we can assume that T captures constant fraction of the Fourier mass, meaning $\sum_{\emptyset\neq S\subseteq T} \widehat{f}(S)^2 \geq \Omega(1)$. Otherwise, the best correlation of f with a k-junta is $o(1) < \varepsilon$ and the task of ε-accurately estimating the distance to the set of k-juntas becomes trivial. Assuming T captures constant fraction of the Fourier mass, Equation (2.1) tells us that we will sample $i \in T$ with constant probability mass. Thus, sampling from this distribution $O(k)$ times means we will have seen most of T up to a small loss in correlation.

To actually estimate these normalized influences, we apply a series of $\log 10k$ random restrictions to our function f (first take 1-random restrictions, then 1/2-random restrictions, then 1/4-random restrictions, and so on), and then show that summing $\widehat{f_{\bar{J}\to z}}(\{i\})^2$ for each of these restrictions is sandwiched between $\mathbf{NInf}_i^{\leq k}[f]$ and $\mathbf{NInf}_i[f]$:

$$\frac{1}{2}\mathbf{NInf}_i^{\leq k}[f] \leq \sum_{i=0}^{\log 10k} \mathop{\mathbf{E}}_{(J,z)\sim\mathcal{R}_{2^{-i}}} \left[\widehat{f_{\bar{J}\to z}}(\{i\})^2\right] \leq 2\mathbf{NInf}_i[f].$$

This would allow us to effectively sample from a proxy distribution that still samples $i \in T$ with constant probability.

We repeat the process iteratively, sampling coordinates one at a time, until we either sampled all of T or sampled a subset $T' \subseteq T$ for which we have that the best junta on T' is almost as correlated with f as the best junta on T. Since the process samples a coordinate in T with constant probability in each round, after $O(k)$ iterations we are likely to succeed, giving us a set U of $O(k)$ coordinates that contains either T or T' (as above). Finally, we show we can estimate, up to a small additive error, the best correlation of a junta-on-U with f, given only approximate oracle access to the coordinates in $\mathcal{S}$. By the above discussion the estimate we get is lower bounded by the best correlation with a k-junta up to a small additive error. It is also upper bounded (trivially) with the best correlation of f with a $O(k)$-junta, since $|U| = O(k)$.

2.1.2 Techniques for Theorem 1.8.1

A limitation of the algorithm we described in the previous subsection is that it only samples one coordinate at a time. In particular, suppose we want to find T exactly, instead of a superset U of T. Then, the naive algorithm would need to consider all subsets of U of size k, estimating the best correlation with a junta on each of them. This gives a $\exp(O(k))$-query algorithm. It would be nicer if we can devise a sampling algorithm that outputs, with constant probability, many coordinates of T at a time. Such a sampling algorithm would reduce the number of possibilities for T in the second stage. In particular, consider the case that the nearest k-junta to f had significant amount of Fourier mass on higher levels, say at level $\approx k$ or maybe $\approx \sqrt{k}$. In this case it would be nice to be able to sample from the Fourier distribution of f, that would give us a large subset of T with constant probability. We note that sampling from the Fourier distribution of a Boolean function is easy for a quantum algorithm but hard for a randomized algorithm. Nevertheless, the (classical) algorithm we describe in this section takes inspiration from this, and samples subsets of size $\sqrt{k}$ according to the Fourier mass of f above level $\sqrt{k}$ of each subset, in time and query complexity $\exp(\widetilde{O}(\sqrt{k}))$.

We will start with the preliminary that we have reduced to the case of only having to consider the coordinates in $\mathcal{S} \subseteq [n]$ with $|\mathcal{S}| \leq O(k/\varepsilon^2)$, using our aforementioned algorithm from the previous section, incurring only a small additive loss in correlation with the closest k-junta. We start with the following definition that generalizes normalized influences of coordinates to normalized influences of sets of coordinates.

Definition 2.1.4. *For a given subset $U \subseteq [n]$, we define its normalized influence as follows:*

$$\mathbf{NInf}_U[f] := \sum_{S\colon U \subseteq S} \frac{\widehat{f}(S)^2}{\binom{|S|}{|U|}}.$$

We also have the natural extension of $\mathbf{NInf}_U^{\leq k}[f] = \sum_{S\colon |S| \leq k, U \subseteq S} \frac{\widehat{f}(S)^2}{\binom{|S|}{|U|}}$, analogous to Definition 2.1.1.

This is a direct generalization of the quantity in Definition 2.1.1. In particular, we consider taking $|U| = \sqrt{k}$. Note there are $2^{\widetilde{O}(\sqrt{k})}$ such U within the coordinates in $\mathcal{S}$, and we can think of these normalized influences as once again defining a sub-probability distribution over subsets of size $\sqrt{k}$. It likely does not sum to 1, but rather sums to $\mathbf{W}_{\mathcal{S}}^{\geq \sqrt{k}}[f] \leq 1$. We show that these normalized influences at exactly level $\sqrt{k}$ can once again be approximated to within a constant factor via a sequence of random restrictions to f:

$$\frac{1}{2}\mathbf{NInf}_U^{\leq k}[f] \leq \sum_{i=0}^{2\sqrt{k}\log 10k} \mathop{\mathbf{E}}_{(J,z)\sim \mathcal{R}_{p^i}} \left[\widehat{f_{\bar{J}\to z}}(U)^2\right] \leq 3\mathbf{NInf}_U[f],$$

where $p = \left(1 - \frac{1}{2\sqrt{k}}\right)$. For more details on this statement, see Theorem 2.3.1.

We are now ready to outline the overall algorithm in Section 2.3. Suppose $T \subseteq \mathcal{S}$ is the subset on which the nearest k-junta (within $\mathcal{S}$) is defined. Our algorithm can then be broken down into two phases:

Phase 1. We get a proxy for $\mathbf{NInf}_U$ for all $|U| = \sqrt{k}$. This is achieved by performing a series of random restrictions to f.

We consider these proxies as a distribution, and sample a constant (this constant is actually dependent on ε, see Section 2.3 for details) number of subsets of size $\sqrt{k}$. With high probability, one of these is in our set of interest T, provided T has a non-negligible amount of Fourier mass above level $\sqrt{k}$.

We don't know which of the subsets we sample are actually in T, so we start a branching process. For each subset we sampled, we restrict f's values in that subset, and recursively sample from sets of size $\sqrt{k}$ using the steps described above. Our branching process will have depth at most $\sqrt{k}$ since at each level we sample $\sqrt{k}$ new coordinates, and T can have at most k relevant coordinates. This phase of our algorithm produces $2^{\widetilde{O}(\sqrt{k})}$ possible subsets of our target set T.

Phase 2. With high probability, one of the branches in the above process will have captured most of the coefficients of T that are relevant above level $\sqrt{k}$ on the Fourier spectrum. Each branch of this process represents a different possibility for what T may be, so for each branch we randomly restrict f so that the coordinates sampled in that branch are fixed, which effectively moves most of the mass of T to levels below $\sqrt{k}$. We then estimate all the Fourier coefficients of this restricted f below level $\sqrt{k}$, allowing us to get an estimate for the closest k-junta on any subset using these estimated coefficients. Each estimation of a Fourier coefficient requires $2^{\widetilde{O}(\sqrt{k})}$-queries to estimate to the desired accuracy, and there are $2^{\widetilde{O}(\sqrt{k})}$ Fourier coefficients to estimate, so overall we make at most $2^{\widetilde{O}(\sqrt{k})}$ queries. From there, for each possible subset of $B \subseteq T$ outputted by phase one, we brute force over all possible subsets of size k containing B, estimating the correlation f has with the closest k-junta on that subset using our estimated Fourier coefficients. This last step takes exponential time in k. We emphasize that while our runtime is exponential in k, our query complexity is only exponential in $\widetilde{O}(\sqrt{k})$.

In the entire above explanation, we have eliminated the dependence on ε for simplicity. We also only consider T for conceptual and analytic simplicity – in reality, we have no idea what T is, and indeed it is exactly what we are looking for. Therefore, more work must be done in order to show that we do not accidentally pick the wrong set, for which our estimates may be inaccurate. To get around this subtle issue, we further apply a noise operator in order to ensure that the significant parts of f lie below level roughly $\sqrt{k}$. We discuss this further in Section 2.3.2.

2.2 Finding a Small(er) Set of Influential Coordinate Oracles

In this section, we detail the process of constructing oracles to coordinates with large low-degree influence. We expand upon the techniques in [DMN19], reducing the number of coordinates one needs to consider to produce a highly correlated k-junta (assuming one exists).

2.2.1 Approximate Oracles to Influential Coordinates

In this subsection we outline and generalize the methods used by [DMN19] to achieve oracle access to coordinates with large low-degree infuence in f. We start with the following definitions from their paper, repeated here for clarity:

Definition 2.2.1 ([DMN19, Def. 3.1])**.** *Let $\mathcal{D}$ be a set of functions mapping $\{\pm 1\}^n$ to $\{\pm 1\}$. We say that $\mathcal{D}$ is an oracle for the coordinates in $\mathcal{S}$ if*

- *for every $g \in \mathcal{D}$, there is some $i \in \mathcal{S}$ such that $g = \pm\mathsf{Dict}_i$; and*
- *for every $i \in \mathcal{S}$, there is some $g \in \mathcal{D}$ such that $g = \pm\mathsf{Dict}_i$.*

In other words, $\mathcal{D}$ is an oracle for $\mathcal{S}$ if $\mathcal{D} = \{\mathsf{Dict}_i : i \in \mathcal{S}\}$ "up to sign".

However, it is not tractable to achieve perfect access to such oracles, so we have to settle for the following weaker notion of approximate oracles:

Definition 2.2.2 ([DMN19, Def. 3.2])**.** *Let $\mathcal{D}$ be a set of functions mapping $\{\pm 1\}^n$ to $\{\pm 1\}$. We say that $\mathcal{D}$ is an ν-oracle for the coordinates in $\mathcal{S}$ if*

- *for every $g \in \mathcal{D}$, there is some $i \in \mathcal{S}$ such that g is ν-close to $\pm\mathsf{Dict}_i$; and*
- *for every $i \in \mathcal{S}$, there is exactly one $g \in \mathcal{D}$ such that g is ν-close to $\pm\mathsf{Dict}_i$; and*
- *For every $g \in \mathcal{D}$, and $\delta > 0$, there is a randomized algorithm that compute $g(x)$ correctly on any $x \in \{\pm 1\}^n$ with probability at least $1-\delta$, using $\mathsf{poly}(k, \log\frac{1}{\delta})$ queries to f.*

Lemma 3.6 in [DMN19] establishes that we can achieve access to a set $\mathcal{D}$ of approximate oracles to $\mathcal{S} \supseteq \{i : \mathbf{Inf}_i^{\leq k}[f] \geq \varepsilon^2/k\}$ of bounded size.

More specifically, we have the following corollary:

Corollary 2.2.3 ([DMN19, Lemma 3.6])**.** *With $\mathsf{poly}(k, \frac{1}{\varepsilon}, \log\frac{1}{\delta}) \cdot \frac{1}{\nu}$ queries to f, we can gain access to an approximate oracle set $\mathcal{D}$ in the sense that for every coordinate i such that $\mathbf{Inf}_i^{\leq k}[f] \geq \frac{\varepsilon^2}{k}$, there exists a $g \in \mathcal{D}$ such that g is ν-close to $\pm\mathsf{Dict}_i$ with probability at least $1-\delta$. Furthermore, $|\mathcal{D}| \leq \mathsf{poly}(k, \frac{1}{\varepsilon}, \log(1/\delta))$.*

We outline the techniques for achieving this lemma in Section 2.2.1, but for the remainder of this chapter we will utilize it as a black box.

For our purposes, we take $\nu = 0.1$ and $\delta = 2^{-\mathsf{poly}(k,\frac{1}{\varepsilon})}$ in all our algorithms. Since we will make much fewer than $2^{\mathsf{poly}(k/\varepsilon)}$-many queries to the coordinate oracles, we can assume that all of our oracles are indeed $\nu = 0.1$ close to dictators/anti-dictators, since by a union bound this is true with high probability.

It is important to note that we do not have a description of which coordinates are influential: from an information theoretic standpoint this would require query complexity dependent on n. What we do have is oracle access to these coordinates in the sense that for all i such that $\mathbf{Inf}_i^{\leq k}[f] \geq \varepsilon^2/k$, there exists $g_i \in \mathcal{D}$ such that $g_i(x) \approx \pm\mathsf{Dict}_i(x)$, that is, $\mathcal{D}$ contains dictators or anti-dictators to every influential coordinate. Using simple techniques of local correction we can simplify this: we need only consider dictators to each coordinate in the oracle. Also, we can convert closeness on average x to high probability correctness for all x (i.e., a worst-case guarantee).

Lemma 2.2.4. *Suppose f is ν-close to $\pm\mathsf{Dict}_i$. For any $x \in \{\pm 1\}^n$, $\mathsf{LocalCorrect}(f, x)$ samples a random $y \sim \{\pm 1\}^n$ and outputs $f(y)f(x \cdot y)$, where $x \cdot y$ is pointwise multiplication. Then,*

$$\forall x : \mathop{\mathbf{Pr}}_{y \sim \{\pm 1\}^n}[\mathsf{LocalCorrect}(f, x) \neq \mathsf{Dict}_i(x)] \leq 2\nu.$$

Proof. Suppose that f is ν close to Dict_i. Then we have $\mathbf{Pr}_{y\sim\{\pm1\}^n}[f(y) \neq \mathsf{Dict}_i(y)] \leq \nu$, and since $x \cdot y$ has the same distribution as y, $\mathbf{Pr}_{y\sim\{\pm1\}^n}[f(x \cdot y) \neq \mathsf{Dict}_i(x \cdot y)] \leq \nu$. Let A be the event that $f(y) \neq \mathsf{Dict}_i(y)$ and let B be the event that $f(x \cdot y) \neq \mathsf{Dict}_i(x \cdot y)$. Clearly if $\mathsf{LocalCorrect}(f, x) \neq \mathsf{Dict}_i(x)$ then at least one of A and B must have occurred (since $\mathsf{Dict}_i(x) = \mathsf{Dict}_i(x \cdot y) \cdot \mathsf{Dict}_i(y)$). Thus, by the union bound, we have

$$\mathop{\mathbf{Pr}}_{y \sim \{\pm 1\}^n}[\mathsf{LocalCorrect}(f, x) \neq \mathsf{Dict}_i(x)] \leq \mathbf{Pr}[A \cup B] \leq \mathbf{Pr}[A] + \mathbf{Pr}[B] \leq 2\nu$$

A similar argument shows that if f is ν close to $-\mathsf{Dict}_i$, then $\mathsf{LocalCorrect}(f, x)$ is not equal to $(-\mathsf{Dict}_i(y))(-\mathsf{Dict}_i(x \cdot y)) = \mathsf{Dict}_i(y)\mathsf{Dict}_i(x \cdot y) = \mathsf{Dict}_i(x)$ with probability at most 2ν. ☐

Given a noisy black box computing h which is ν-close to $g = \pm\mathsf{Dict}_i$, local correction will compute Dict_i with high probability, on every input x. Critically, we can treat potentially faulty $\pm\mathsf{Dict}_i$ oracles as correct Dict_i oracles provided suitably many repetitions.

Corollary 2.2.5. *If f is ν-close to $\pm\mathsf{Dict}_i$ for $\nu = 0.1$, then repeating $\mathsf{LocalCorrect}(f, x)$ independently $\mathsf{poly}(k, 1/\varepsilon)$ times and taking the majority outcome results in an incorrect value for $\mathsf{Dict}_i(x)$ with probability at most $2^{-\mathsf{poly}(k,1/\varepsilon)}$.*

Proof. Clear from applying the first bound in Fact 1.9.1 with $N = O(\mathsf{poly}(k/\varepsilon))$ and $\eta = (1 - 2\nu - 0.5) = 0.3$ in this case. ☐

We also show that restricting our attention to $\mathcal{S}$ we have not lost more than ε in the best correlation of f with a k-junta. This is proved in the following claim.

Claim 2.2.6. *Let $f : \{\pm 1\}^n \to \{\pm 1\}$ and let $g : \{\pm 1\}^n \to \{\pm 1\}$ be a k-junta on U. Let $\tau > 0$. Take*

$$S = \left\{ i \in U \;\middle|\; \mathbf{Inf}_i^{\leq k}[f] \geq \tfrac{\tau^2}{k} \right\}$$

Then, there is a junta on S with correlation at least $\mathbf{E}[fg] - \tau$ with f.

Proof. To prove this claim, we define a function on the set S such that the loss in correlation is at most τ. Consider:

$$g'(x) = g_{\mathsf{avg},S}(x) = \mathop{\mathbf{E}}_{y}[g(y) | y_S = x_S]$$

First, we note g' is a function over only the variables in S. Second, it is bounded in $[-1, 1]$, so it is not quite Boolean, but it can be randomized rounded to a Boolean function, with the expected correlation with f equaling $\mathbf{E}[fg']$. Thus, it suffices to show that $\mathbf{E}[fg'] \geq \mathbf{E}[fg] - \tau$ to deduce that there exists a randomize rounding of g' to a Boolean function g'' with $\mathbf{E}[fg''] \geq \mathbf{E}[fg] - \tau$. We also recall that

$$\widehat{g'}(T) = \begin{cases} \widehat{g}(T) & \text{if } T \subseteq S \\ 0 & \text{otherwise} \end{cases}$$

We thus have:

$$|\mathbf{E}[fg] - \mathbf{E}[fg']| = \left| \sum_{\substack{T \not\subseteq S \\ T \subseteq U}} \widehat{f}(T)\widehat{g}(T) \right| \leq \sqrt{\sum_{\substack{T \not\subseteq S \\ T \subseteq U}} \widehat{f}(T)^2} \leq \sqrt{\sum_{i \in U \setminus S} \sum_{\substack{T \ni i \\ T \subseteq U}} \widehat{f}(T)^2}$$
$$\leq \sqrt{\sum_{i \in U \setminus S} \mathbf{Inf}_i^{\leq k}(f)} \leq \sqrt{k \cdot \frac{\tau^2}{k}} = \tau \qquad \square$$

Finally, the below corollary summarizes what we have achieved in this section.

Corollary 2.2.7. *With* $\mathsf{poly}(k, \frac{1}{\varepsilon}, \log \frac{1}{\delta})$ *queries to* f*, we can gain access to an approximate oracle set* $\mathcal{D}$ *for a set of coordinates* $\{i : \mathbf{Inf}_i^{\leq k} \geq \frac{\varepsilon^2}{k}\} \subseteq \mathcal{S} \subseteq [n]$*. Moreover, these coordinates and oracles satisfy the following properties.*

- *For every coordinate* $i \in \mathcal{S}$*, there exists a* $g \in \mathcal{D}$ *such that* g *is* 0.1*-close to* Dict_i *with probability at least* $1 - \delta$*.*
- $\mathsf{dist}(f, \mathcal{J}_{n,k}) - \mathsf{dist}(f, \mathcal{J}_{\mathcal{S},k}) \leq \varepsilon$*.*
- $|\mathcal{S}| \leq \mathsf{poly}(k, 1/\varepsilon, \log(1/\delta))$*.*
- *For any algorithm* A *that uses at most* q *queries to* $\mathcal{D}$*, we can use* $\mathsf{LocalCorrect}$ *from Lemma 2.2.4 with error* δ/q *to assume that we actually have perfect access to each coordinate oracle, up to an additive loss of* δ *in confidence and a multiplicative overhead of* $\mathsf{poly}(\log(q/\delta))$ *in query complexity.*

Proof. The first and the third bullet point follow from Corollary 2.2.3. The second bullet point follows from Claim 2.2.6. To achieve the last point, we can use Corollary 2.2.5 every time we make a "query" to an oracle in our algorithm. Thus every "query" to an oracle $g \approx \pm\mathsf{Dict}_i$ at x involves $\mathsf{poly}(\log(q/\delta))$ many repetitions of $\mathsf{LocalCorrect}(g, x)$, which results in an incorrect value with probability at most $\delta/2q$, as noted above. Recall that Corollary 2.2.3 guarantees that we can output $g(x)$ correctly with probability $1 - \delta/2q$ with only a $\mathsf{poly}(k, \log(q/\delta))$ queries to f. Since we only ever make at most q queries to our coordinate oracles, we can assume that $\mathsf{LocalCorrect}(g, x) = \mathsf{Dict}_i(x)$ in all queries. This happens with probability at least $1 - \delta$ by the union bound. $\square$

Therefore, for the rest of this thesis, we will assume that we have oracle access to *exact dictators.*

2.2.2 Implicit Access to an Underlying Junta

An important consequence of having coordinate oracles is that it allows us to reduce the input size of the function dramatically. Suppose $f : \{\pm 1\}^n \to \{\pm 1\}$ and we have $\mathcal{D} = \{g_1, \dots, g_{k'}\}$ are randomized algorithms that for any $x \in \{\pm 1\}^n$ output $g_i(x) = \mathsf{Dict}_{j_i}(x) = x_{j_i}$. We have that $j_1, \dots, j_{k'} \in [n]$ are a set of k' distinct coordinates. Let $U = \{j_1, \dots, j_{k'}\}$. We want to get access to the following function: $g(x_1, \dots, x_{k'}) = \mathbf{E}[f(y) | y_{j_1} = x_1, y_{j_2} = x_2, \dots, y_{j_{k'}} = x_{k'}]$. More precisely, given $x_1, \dots, x_{k'}$ we want to sample uniformly from all $y \in \{\pm 1\}^n$ that satisfy $y_{j_1} = x_1, y_{j_2} = x_2, \dots, y_{j_{k'}} = x_{k'}$ and apply f on this y.

The following algorithm that runs in $\mathsf{poly}(k, \log(1/\delta))$ time samples y from such a distribution.

Algorithm 5: Sampling a uniformly random input consistent with the oracles' values

```
Input: $f$ (target function), $\mathcal{D} = \{g_1, \ldots, g_{k'}\}$ (coordinate oracles),
       $(x_1, \ldots, x_{k'}) \in \{\pm 1\}^{k'}$
Output: A vector $y \in \{\pm 1\}^n$ with $(g_1(y), \ldots, g_{k'}(y)) = (x_1, \ldots, x_{k'})$
1 Sample $y \sim \{\pm 1\}^n$ and let $z \in \{\pm 1\}^{k'}$ be the vector of evaluations of $\{g_1, \ldots, g_{k'}\}$
    on $y$;
2 while $z \neq x$ do
3     repeat
4         Let $y'$ be a copy of $y$, but flip each bit independently with probability $\frac{1}{k'}$;
5         Let $z'$ be the vector of evaluations of $\{g_1, \ldots, g_{k'}\}$ on $y'$;
6     until $\mathsf{dist}(x, z') < \mathsf{dist}(x, z)$
7     $y = y'$;
8     $z = z'$;
9 return $y$
```

Theorem 2.2.8. *Algorithm 5 with probability* $1 - \delta$ *runs in time* $\mathsf{poly}(k', \log(1/\delta))$.

Proof. We focus on the number of iterations of the inner repeat loop. Given (y, z) with $z \neq x$ we analyze the time it takes to find a (y', z') with $\mathsf{dist}(z', x) < \mathsf{dist}(z, x)$. Since $x \neq z$ without loss of generality we can assume that $x_1 \neq z_1$.

To get (y', z') with $\mathsf{dist}(z', x) < \mathsf{dist}(z, x)$, it suffices to sample a vector y' with $y'_{j_1} = x_1$ and $y'_{j_2} = y_{j_2}, y'_{j_3} = y_{j_3}, \ldots, y'_{j_{k'}} = y_{j_{k'}}$. Indeed, since we are flipping each coordinate with probability $1/k'$ the probability of sampling such a y' is exactly $1/k' \cdot (1 - 1/k')^{k'-1} \geq 1/(ek')$. Thus, we get that the runtime of the repeat loop is stochastically dominated by a geometric random variable with success probability $1/(ek')$. Thus with probability at least $1 - \delta/k'$, it finishes after $O(k' \cdot \log(k'/\delta))$ iterations. We run the inner repeat loop at most k'-times, thus by union bound, with probability at least $1 - \delta$ the entire process end after at most $O(k'^2 \cdot \log(k'/\delta))$ executions of line 5. We note that execution line 5 actually requires k' queries to $g_1, \ldots, g_{k'}$, each of them takes $\mathsf{poly}(k) = \mathsf{poly}(k')$ time. thus overall, with probability at least $1 - \delta$, our algorithm run in time $\mathsf{poly}(k', \log(1/\delta))$. □

Theorem 2.2.9. *Algorithm 5 samples uniformly from the set of inputs* $\{y' : (g_1(y'), \ldots, g_{k'}(y')) = (x_1, \ldots, x_{k'})\}$.

Proof. Let $U = \{j_1, \ldots, j_{k'}\}$ be the set of coordinates for which $\{g_1, \ldots, g_{k'}\}$ are oracles to. Algorithm 5 certainly samples a vector y with $y_{j_1} = x_1, \ldots, y_{j_{k'}} = x_{k'}$. We want to show additionally that Algorithm 5 samples $y_{\overline{U}}$ uniformly at random. In fact, at any point in the algorithm the distribution over $y_{\overline{U}}$ is uniform. This is clearly true in the first step where $y \sim \{\pm 1\}^n$, and remains true along the algorithm as we apply independent noise to coordinates in $\overline{U}$ and decide whether to apply the noise or not according to the value of y_U which is independent of $y_{\overline{U}}$. □

We will consider algorithms computing non-Boolean function like $g = f_{\mathsf{avg},S}$ for some subset $S \subseteq [n]$. Note that g is a function whose range in $[-1, 1]$, but not necessarily a Boolean function.

Theorem 2.2.10 (Formal version of Theorem 1.8.4)**.** *Let $f : \{\pm 1\}^n \to \{\pm 1\}$, $\mathcal{D} = \{g_1, \dots, g_{k'}\}$ be a set of coordinate oracles. Let g be a function from $\{\pm 1\}^{k'} \to [-1, 1]$ defined by $g(x) = \mathbf{E}[f(y) | g_1(y) = x_1, \dots, g_{k'}(y) = x_{k'}]$. Then g has a randomized algorithm in the sense of Definition 1.9.9 computing it that runs in expected time $\mathsf{poly}(k')$.*

Proof. Given $x = (x_1, \dots, x_{k'})$ apply Algorithm 5 on f, $\mathcal{D}$ and x to get a vector $y \in \{\pm 1\}^n$. Return $f(y)$. It is clear that since y is a uniform input subject to $g_1(y) = x_1, \dots, g_{k'}(y) = x_{k'}$ that our algorithm is a randomized algorithm for g. □

2.2.3 Influential Coordinate Oracles

As above, denote as $\mathcal{S}$ the superset of the low-degree influential coordinates of f, and $\mathcal{D}$ as the set of approximate oracles to said coordinates, obtained via Corollary 2.2.3 with parameter $\nu = 0.1$. As we discussed in Section 2.2.1, we assume (with a small loss in error probability, and a small multiplicative factor on query complexity) that we have exact access to dictators for each influential coordinate. We work towards proving the following improved version of a corollary that appeared in [DMN19]:

The idea will be to take $\mathcal{D}$, a set of $k' = \mathsf{poly}\left(k, \frac{1}{\varepsilon}\right)$ coordinate oracles, and somehow "prune" it down to a set $\mathcal{D}'$ of at most $O(\frac{k}{\varepsilon^2})$ coordinate oracles, such that that the loss in the most correlated junta on this smaller set of coordinates is at most ε

$$\max_{g \in \mathcal{J}_{\mathcal{D},k}} \mathbf{E}[fg] - \max_{g \in \mathcal{J}_{\mathcal{D}',k}} \mathbf{E}[fg] \le \varepsilon.$$

2.2.4 Reducing the Number of Oracles to Consider

Starting with a set of $\mathsf{poly}(k/\varepsilon)$ set of oracles $\mathcal{D}$ for a set $\mathcal{S}$ containing the influential coordinates of f, our goal in this section is to prune the number of oracles to $O(k/\varepsilon^2)$ in a way that incurs only a small loss in correlation with the nearest k-junta. [DMN19] achieved their theorem by noting that applying a standard noise operator to f did not affect its proximity to the nearest k-junta significantly, while also guaranteeing that at most $\frac{k^2}{\varepsilon^2}$ coordinates could have large influence. They then were able to estimate the influence of every coordinate in $\mathcal{D}$ despite only having (approximate) oracle access to the influential coordinates, and thus were able to determine which oracles were actually oracles to influential coordinates, of which there were less than k^2/ε^2.

Our approach, as explained at a high level in Section 2.1, is to estimate the normalized influence of each coordinate in $\mathcal{S}$, which is done via a sequence of random restrictions to f. In words, the below algorithm estimates for each coordinate $i \in \mathcal{S}$ the quantity $\lambda_i^{\approx 2^d} = \mathbf{E}_{(J,z) \sim \mathcal{R}_{2^{-d}}}[\widehat{f_{\bar{J} \to z}}(\{i\})^2]$, where $(J, z) \sim \mathcal{R}_{2^{-d}}$ parameterize a 2^{-d}-random restriction to f. Then, λ_i is defined to be sum over a series of random restrictions $d = 0, ..., \log 10k$ of $\lambda_i^{\approx 2^d}$. The core idea of our algorithm is that this sum over Fourier coefficients on the first level of restricted versions of f is a proxy for $\mathbf{NInf}_i[f]$. In other words, we have the following theorem:

Theorem 2.2.11. *Let $f : \{\pm 1\}^{k'} \to \mathbb{R}$, where $k' = |\mathcal{D}|$. Let $i \in [k']$. Let*

$$\lambda_i[f] = \sum_{m=0}^{\log(10k)} \lambda_i^{\approx 2^m}[f], \qquad \textit{where} \qquad \lambda_i^{\approx 2^m}[f] = \mathop{\mathbf{E}}_{(J,z) \sim \mathcal{R}_{2^{-m}}}[\widehat{f_{\bar{J} \to z}}(\{i\})^2].$$

Then, $\frac{1}{2}\mathbf{NInf}_i^{\leq k}[f] \leq \lambda_i[f] \leq 2\mathbf{NInf}_i[f]$.

We postpone the proof of Theorem 2.2.11 to Section 2.2.5. The definition of λ_i naturally gives rise to an algorithm for estimating λ_i that we present next. The algorithm would return for each $i \in [k']$ an estimate $\widetilde{\lambda_i}$ that would be close to λ_i with high probability.

Algorithm 6: Estimating λ_i

```
   Input: $f : \{\pm 1\}^{k'} \to [-1, 1]$ along with randomized algorithm A computing $f$
          (recall Def. 1.9.9). Parameters $1-\delta$ (confidence), $\varepsilon$ (additive error) and $k$.
   Output: Estimates $(\widetilde{\lambda_1}, \dots, \widetilde{\lambda_{k'}})$ for $(\lambda_1, \dots, \lambda_{k'})$.
 1 Let $m = \mathsf{poly}(k, k', 1/\varepsilon, \log(1/\delta))$
 2 Initialize $\widetilde{\lambda_i} = 0$ for all $i \in [k']$;
 3 for $d = 0$ to $\log 10k$ do
 4     Initialize $\widetilde{\lambda}_i^{\approx 2^d} = 0$ for all $i \in [k']$;
 5     repeat $m$ times
 6         Let $(J, z) \sim \mathcal{R}_{2^{-d}}$ be a $2^{-d}$-random restriction.
 7         Estimate $\widehat{f_{\bar{J}\to z}}(\{j\})$ for all $j \in J$ up to additive error $\frac{\varepsilon}{6\log(10k)}$ with
            probability $1 - \delta/\mathsf{poly}(k, k', m)$ using Claim 1.9.10 and algorithm A.
 8         Denote by $\widetilde{f_{\bar{J}\to z}}(\{j\})$ the estimated Fourier coefficient.
 9         Update $\widetilde{\lambda}_j^{\approx 2^d} = \widetilde{\lambda}_j^{\approx 2^d} + \widetilde{f_{\bar{J}\to z}}(\{j\})^2$ for all $j \in J$.
10     Let $\widetilde{\lambda}_i^{\approx 2^d} = \widetilde{\lambda}_i^{\approx 2^d}/m$ for all $i \in [k']$;
11 Let $\widetilde{\lambda_i} = \sum_d \widetilde{\lambda}_i^{\approx 2^d}$;
12 return $(\widetilde{\lambda_1}, \widetilde{\lambda_2}, \dots, \widetilde{\lambda_{k'}})$
```

Lemma 2.2.12. *With probability at least* $1-\delta$ *we have that for all* $i \in [k']$ *it holds that* $|\widetilde{\lambda_i} - \lambda_i| \leq \varepsilon$.

Proof. If $j \notin J$ the Fourier coefficient of $\widehat{f_{\bar{J}\to z}}$ is 0 and so our estimate is correct in that case. In the case $j \in J$, each estimation of the Fourier coefficient is correct up to additive error $\eta = \varepsilon/6\log(10k)$ with probability at least $1 - \delta/\mathsf{poly}(k, k', m)$. Thus, we get that $\widetilde{f_{\bar{J}\to z}}(\{j\})^2 = (\widehat{f_{\bar{J}\to z}}(\{j\}) \pm \eta)^2 = \widehat{f_{\bar{J}\to z}}(\{j\})^2 \pm 2\eta|\widehat{f_{\bar{J}\to z}}(\{j\})| \pm \eta^2 = \widehat{f_{\bar{J}\to z}}(\{j\})^2 \pm 3\eta$. Furthermore, we have that $\mathbf{E}_{(J,z)\sim\mathcal{R}_{2^{-d}}}[\widehat{f_{\bar{J}\to z}}(\{j\})^2] = \lambda_j^{\approx 2^d}$, thus by Fact 1.9.1 we have that the empirical mean of $m = \mathsf{poly}(1/\varepsilon, \log(k), \log(k'), \log(1/\delta))$ copies of $\widetilde{f_{\bar{J}\to z}}(\{j\})^2$ is within additive error $\varepsilon/(2\log(10k))$ from $\lambda_j^{\approx 2^d}$ with probability at least $1 - \delta/(k' \log(10k))$. By union bound, all these estimates are within the error bound, and we get that $|\widetilde{\lambda}_j^{\approx 2^d} - \lambda_j^{\approx 2^d}| \leq 3\eta + \varepsilon/(2\log(10k)) \leq \varepsilon/(\log(10k))$. Overall, we get that $|\widetilde{\lambda}_j - \lambda_j| \leq \varepsilon$ for all $j \in [k']$ with probability at least $1-\delta$. □

With Algorithm 6 in hand, we are ready to present the pruning procedure.

```
Algorithm 7: Reduce Number of Oracles
  Input: $f$ (target function), $\mathcal{D}$ (influential coordinate oracles, where $\mathcal{D}$ are oracles
         for $\mathcal{S}$). Parameters $\varepsilon$ and $\delta$.
  Output: A subset $\mathcal{D}' \subseteq \mathcal{D}$ of size $O(\frac{k}{\varepsilon^2})$ such that we lose at most $\varepsilon$ in correlation
          with $f$.
1 Initialize $\mathcal{D}' = \emptyset$;
2 Let $m = O((k + \log(1/\delta))/\varepsilon^2)$
3 repeat $m$ times
4     Let $\{g_1, \ldots, g_{k'}\} = \mathcal{D} - \mathcal{D}'$, and $\{g_{k'+1}, \ldots, g_{|\mathcal{D}|}\} = \mathcal{D}'$
5     Sample $z \in \{\pm 1\}^{|\mathcal{D}'|}$. Let $f' : \{\pm 1\}^{k'} \to \mathbb{R}$ be the function defined by
        $f'(x_1, \ldots, x_{k'}) = \mathop{\mathbf{E}}_{y \sim \{\pm 1\}^n}[f(y) | g_1(y) = x_1, \ldots, g_{k'}(y) = x_{k'}, g_{k'+1}(y) = z_1, \ldots, g_{k'+|\mathcal{D}'|}(y) = z_{|\mathcal{D}'|}]$.
      and let A be the randomized algorithm for $f'$ from Theorem 2.2.10.
6     Apply Algorithm 6 on $f'$ using the randomized algorithm A for $f'$ with
      confidence $1 - \frac{\delta}{2m}$ and accuracy $\frac{\varepsilon^2}{48 \cdot |\mathcal{S}|}$ $\implies \widetilde{\lambda} = (\widetilde{\lambda}_1, \ldots, \widetilde{\lambda}_{k'})$.
7     Let our distribution $P$ be defined by $\widetilde{\lambda}$, normalized appropriately.
8     Sample $i \sim P$, and add $g_i$ to $\mathcal{D}'$.
9 return $\mathcal{D}'$
```

Lemma 2.2.13. *With probability at least $1 - \delta$, Algorithm 7 returns a set of oracles $\mathcal{D}'$ to a subset of coordinates $\mathcal{S}' \subseteq \mathcal{S}$, such that*

$$\max_{g \in \mathcal{J}_{\mathcal{S},k}} \mathbf{E}[fg] - \max_{g \in \mathcal{J}_{\mathcal{S}',k}} \mathbf{E}[fg] \leq \varepsilon.$$

To prove Lemma 2.2.13, which tells us our algorithm succeeds and directly implies Theorem 1.8.3, we will need a few more lemmas.

We denote the event $\mathcal{E}$ that in the entire execution of Algorithm 7 all $\widetilde{\lambda}_i$ were $\varepsilon^2/(48 \cdot |\mathcal{S}|)$ close to the real λ_i. We note that by union bound this event happens with probability at least $1 - \delta/2$.

Suppose T is the (unknown) set of k oracles for which the best-k junta approximating f is a junta on T. We want to show that our algorithm either samples all the coordinates in T, or it samples a subset T' of T that captures all but $\varepsilon^2/4$ of the Fourier mass of f on T.

Claim 2.2.14. *Assume the event $\mathcal{E}$ happens. Then, with probability at least $1 - \delta/2$, after m iterations, we will have either:*

1. *sampled i for all $i \in T$, our target set;*
2. *sampled i for all $i \in T' \subseteq T$, where $\sum_{S \subseteq T'} \widehat{f}(S)^2 \geq \sum_{S \subseteq T} \widehat{f}(S)^2 - \varepsilon^2/4$.*

Proof. In each iteration, assume we have not yet satisfied either items. Let V be the subset of coordinates in T that we have not yet sampled. Let $T' = T \setminus V$. By assumption,

$$\varepsilon^2/4 < \sum_{S \subseteq T} \widehat{f}(S)^2 - \sum_{S \subseteq T'} \widehat{f}(S)^2 = \sum_{S \subseteq T : S \cap V \neq \emptyset} \widehat{f}(S)^2.$$

Let $\mathcal{S}'' = \mathcal{S} \setminus \mathcal{S}'$. We have that $|\mathcal{S}''| = k'$. Now note that up to relabeling of coordinates f' from Algorithm 7 is the same as $(f_{\mathsf{avg},\mathcal{S}})_{\mathcal{S}' \to z}$, where z was randomly chosen. For brevity, denote by $f_z = (f_{\mathsf{avg},\mathcal{S}})_{\mathcal{S}' \to z}$. Note that for any fixed z, f_z is a function that depends only on the coordinates in $\mathcal{S}''$.

By Fact 1.9.7, we have

$$\mathop{\mathbf{E}}_{z}\left[\sum_{\emptyset \neq S \subseteq V} \widehat{f_z}(S)^2\right] = \sum_{R : \emptyset \neq (R \cap \mathcal{S}'') \subseteq V} \widehat{f_{\mathsf{avg},\mathcal{S}}}(R)^2 = \sum_{\substack{R \subseteq \mathcal{S}: \\ \emptyset \neq (R \cap \mathcal{S}'') \subseteq V}} \widehat{f}(R)^2 \geq \sum_{R \subseteq T : R \cap V \neq \emptyset} \widehat{f}(R)^2$$
$$> \varepsilon^2/4. \tag{2.2}$$

Next, by applying Theorem 2.2.11, for any fixed z, we have

$$\sum_{i \in V} \lambda_i[f_z] \geq \frac{1}{2} \sum_{i \in V} \mathbf{NInf}_i^{\leq k}[f_z] \geq \frac{1}{2} \sum_{\emptyset \neq S \subseteq V} \widehat{f_z}(S)^2.$$

By the assumption that $\mathcal{E}$ happens, the $\widetilde{\lambda}_i$ are $\frac{\varepsilon^2}{48 \cdot |\mathcal{S}|}$-accurate, and we get that

$$\sum_{i \in V} \widetilde{\lambda}_i[f_z] \geq \frac{1}{2} \sum_{\emptyset \neq S \subseteq V} \widehat{f_z}(S)^2 - \frac{\varepsilon^2}{48 \cdot |\mathcal{S}|} \cdot |V| \geq \frac{1}{2} \sum_{\emptyset \neq S \subseteq V} \widehat{f_z}(S)^2 - \frac{\varepsilon^2}{48}.$$

On the other hand by applying Theorem 2.2.11 again we see that

$$\sum_{i \in \mathcal{S}''} \lambda_i[f_z] \leq 2 \cdot \sum_{i \in \mathcal{S}''} \mathbf{NInf}_i[f_z] = 2 \cdot \mathbf{Var}[f_z] \leq 2$$

and thus $\sum_{i \in \mathcal{S}''} \widetilde{\lambda}_i[f] \leq 2 + k' \cdot \frac{\varepsilon^2}{48 \cdot |\mathcal{S}|} \leq 2 + \frac{\varepsilon^2}{48} \leq 3$ (under the assumption that $\mathcal{E}$ happens). Overall, the probability to sample an element from V is at least

$$\frac{1}{3} \cdot \left(\frac{1}{2} \sum_{\emptyset \neq S \subseteq V} \widehat{f_z}(S)^2 - \frac{\varepsilon^2}{48}\right) = \frac{1}{6} \sum_{\emptyset \neq S \subseteq V} \widehat{f_z}(S)^2 - \frac{\varepsilon^2}{3 \cdot 48}$$

By taking expectation over z, and using Equation (2.2) we see that the probability to sample an element from V overall is at least

$$\mathop{\mathbf{E}}_{z}\left[\frac{1}{6} \sum_{\emptyset \neq S \subseteq V} \widehat{f_z}(S)^2 - \frac{\varepsilon^2}{3 \cdot 48}\right] \geq \frac{1}{6} \cdot \frac{\varepsilon^2}{4} - \frac{\varepsilon^2}{3 \cdot 48} > \frac{\varepsilon^2}{30}.$$

We get that in each iteration as long as we don't satisfy Items (1) and (2) above, we sample an element from $i \in T$ with probability at least $\varepsilon^2/30$. By repeating the process $m = O(\frac{k + \log(1/\delta)}{\varepsilon^2})$ times we would sample all of T, or get stuck at some T' satisfying Item (2), with probability at least $1 - \delta/2$, using Fact 1.9.1. $\square$

Next, we show that finding T' is almost as good as finding T in the sense that the best correlation by juntas-on-T' with f is up to small additive error the best correlation by juntas-on-T with f.

Lemma 2.2.15. *Suppose we have some subset T such that $\sum_{S\subseteq T}\widehat{f}(S)^2 = c$, and we then identified a subset $T' \subseteq T$ such that $\sum_{S\subseteq T'}\widehat{f}(S)^2 \geq c - \frac{\varepsilon^2}{4}$. Then*

$$\left|\max_{g\in\mathcal{J}_{T,k}} \mathbf{E}[fg] - \max_{g\in\mathcal{J}_{T',k}} \mathbf{E}[fg]\right| \leq \varepsilon$$

Proof. We know that $\operatorname{argmax}_{g\in\mathcal{J}_{T,k}} E[fg] = \mathsf{sgn}(f_{\mathsf{avg},T})$ and similarly $\operatorname{argmax}_{g\in\mathcal{J}_{T',k}} E[fg] = \mathsf{sgn}(f_{\mathsf{avg},T'})$. Then we have that

$$\begin{aligned}
\left|\max_{g\in\mathcal{J}_{T,k}} \mathbf{E}[fg] - \max_{g\in\mathcal{J}_{T',k}} \mathbf{E}[fg]\right| &= \mathbf{E}[f(x)(\mathsf{sgn}(f_{\mathsf{avg},T}(x_T)) - \mathsf{sgn}(f_{\mathsf{avg},T'}(x_{T'})))] \\
&= \mathop{\mathbf{E}}_{x_T}\left[\mathop{\mathbf{E}}_{x_{\overline{T}}}[f(x_T, x_{\overline{T}})]\,(\mathsf{sgn}(f_{\mathsf{avg},T}(x_T)) - \mathsf{sgn}(f_{\mathsf{avg},T'}(x_{T'})))\right] \\
&= \mathop{\mathbf{E}}_{x_T}\left[f_{\mathsf{avg},T}(x_T)\,(\mathsf{sgn}(f_{\mathsf{avg},T}(x_T)) - \mathsf{sgn}(f_{\mathsf{avg},T'}(x_{T'})))\right] \\
&\leq 2\mathop{\mathbf{E}}_{x_T}\left[|f_{\mathsf{avg},T}(x_T) - f_{\mathsf{avg},T'}(x_{T'})|\right] \\
&\qquad\text{(Since } z(\mathsf{sgn}(z) - \mathsf{sgn}(z')) \leq 2|z-z'| \text{ for all } z, z' \in \mathbb{R}) \\
&\leq 2\sqrt{\mathop{\mathbf{E}}_{x_T}\left[\left(f_{\mathsf{avg},T}(x_T) - f_{\mathsf{avg},T'}(x_{T'})\right)^2\right]} \\
&= 2\sqrt{\sum_{S\subseteq T}\widehat{f}(S)^2 - 2\sum_{S\subseteq T'}\widehat{f}(S)^2 + \sum_{S\subseteq T'}\widehat{f}(S)^2} \\
&\leq 2\sqrt{\frac{\varepsilon^2}{4}} = \varepsilon\,.
\end{aligned}$$

□

Proof of Lemma 2.2.13. Let g be the k-junta that maximizes $\mathbf{E}[fg]$ among all k-juntas on $\mathcal{S}$. Let T be the set of variables on which g depends. By Claim 2.2.14 we either sample oracles to all of T or to a subset T' for which

$$\sum_{S\subseteq T'}\widehat{f}(S)^2 \geq \sum_{S\subseteq T}\widehat{f}(S)^2 - \varepsilon^2/4.$$

In the second case, by Lemma 2.2.15, we incur a loss in correlation of at most ε with our nearest k-junta. In the first case, we lose no correlation with the closest k-junta, and by a union bound our probability of failure is at most δ. □

The above concludes the proof of Lemma 2.2.13. Finally, Theorem 1.8.3 is implied by Lemma 2.2.13, as shown below.

Theorem 2.2.16 (Theorem 1.8.3, restated)**.** *Let $\varepsilon > 0$, $k \in \mathbb{N}$, and $k' = C(k/\varepsilon^2)$ for some universal constant C. Then, there exists an algorithm that given f, k, ε makes at most poly$(k, 1/\varepsilon)$ queries to f and returns a number α such that with probability at least* 0.99

1. $\alpha \leq \max_{g\in\mathcal{J}_{n,k'}} \mathbf{E}[fg] + O(\varepsilon)$
2. $\alpha \geq \max_{g\in\mathcal{J}_{n,k}} \mathbf{E}[fg] - O(\varepsilon)$

Proof. Set $\delta = 2^{-\mathsf{poly}(k,1/\varepsilon)}$. We first apply Corollary 2.2.3 from [DMN19]. This gives us $\mathsf{poly}(k, \frac{1}{\varepsilon}, \log(1/\delta)) = \mathsf{poly}(k/\varepsilon)$ coordinate oracles $\mathcal{D}$ to coordinates $\mathcal{S}$ that includes all coordinates i with $\mathbf{Inf}_i^{\leq k}[f] \geq \frac{\varepsilon^2}{k}$. By Claim 2.2.6 we see that

$$\max_{g \in \mathcal{J}_{\mathcal{S},k}} \mathbf{E}[fg] \geq \max_{g \in \mathcal{J}_{n,k}} \mathbf{E}[fg] - \varepsilon$$

Next, we apply Algorithm 7 to get a subset $\mathcal{D}' \subseteq \mathcal{D}$ to coordinates $\mathcal{S}' \subseteq \mathcal{S}$ such that with high probability

$$\max_{g \in \mathcal{J}_{\mathcal{S}',k}} \mathbf{E}[fg] \geq \max_{g \in \mathcal{J}_{\mathcal{S},k}} \mathbf{E}[fg] - \varepsilon$$

We take α to be the estimation of the correlation of the best junta on $\mathcal{S}'$ with f. By Claim 1.9.2 we have that $\max_{g \in \mathcal{J}_{\mathcal{S}'}} \mathbf{E}[fg] = \mathbf{E}[|f_{\mathsf{avg},\mathcal{S}'}(x)|]$. To estimate the latter, we use a randomized algorithm that computes $f_{\mathsf{avg},\mathcal{S}'}$ given by Theorem 2.2.10. We randomly sample $O(1/\varepsilon^2)$ many values for x and estimate for each of them $|f_{\mathsf{avg},\mathcal{S}'}(x)|$ up to additive error $\varepsilon/2$ via the randomized algorithm with expected value $f_{\mathsf{avg},\mathcal{S}'}(x)$.

Assume that α is a ε-additive approximation to $\max_{g \in \mathcal{J}_{\mathcal{S}'}} \mathbf{E}[fg]$. In this case, we claim that α satisfies both items from the theorem's statement. Indeed,

1. $\alpha \leq \max_{g \in \mathcal{J}_{\mathcal{S}'}} \mathbf{E}[fg] + \varepsilon \leq \max_{g \in \mathcal{J}_{n,k'}} \mathbf{E}[fg] + \varepsilon$.
2. $\alpha \geq \max_{g \in \mathcal{J}_{\mathcal{S}'}} \mathbf{E}[fg] - \varepsilon \geq \max_{g \in \mathcal{J}_{\mathcal{S}',k}} \mathbf{E}[fg] - \varepsilon \geq \max_{g \in \mathcal{J}_{\mathcal{S},k}} \mathbf{E}[fg] - 2\varepsilon \geq \max_{g \in \mathcal{J}_{n,k}} \mathbf{E}[fg] - 3\varepsilon$.

Next, we analyze the number of queries of our algorithm. Obtaining the initial set of coordinate oracles $\mathcal{D}$ takes $\mathsf{poly}(k, 1/\varepsilon, \log(1/\delta)) = \mathsf{poly}(k, 1/\varepsilon)$ queries. Then, we go on to run Algorithm 7 that makes $m = O((k + \log(1/\delta))/\varepsilon^2)$ iterations, each making $\mathsf{poly}(k, 1/\varepsilon, \log(1/\delta))$ queries. Next, to estimate $\mathbf{E}[|f_{\mathsf{avg},\mathcal{S}'}(x)|]$ we require $\mathsf{poly}(1/\varepsilon)$ samples from randomized algorithm for $f_{\mathsf{avg},\mathcal{S}'}(x)$ each such sample translate to $\mathsf{poly}(k, 1/\varepsilon)$ samples to f. Finally, we note that each "query" to an oracle incurs an overhead of $\mathsf{poly}(\log(k, 1/\varepsilon))$ queries to f along with an $o(1)$ additive loss in confidence by Corollary 2.2.7. Overall, we make $\mathsf{poly}(k, 1/\varepsilon)$ queries. ☐

2.2.5 Proof of Theorem 2.2.11

We now present the proof of Theorem 2.2.11.

Proof of Theorem 2.2.11. We express λ_i in terms of the Fourier spectrum of f. Using Fact 1.9.7,

$$\begin{aligned}
\lambda_i &= \sum_{m=0}^{\log(10k)} \sum_{S: S \ni i} \widehat{f}(S)^2 \cdot \mathop{\mathbf{Pr}}_{J \subseteq_{2^{-m}} [k']}[S \cap J = \{i\}] \\
&= \sum_{m=0}^{\log(10k)} \sum_{S: S \ni i} \widehat{f}(S)^2 \cdot \mathop{\mathbf{Pr}}_{J \subseteq_{2^{-m}} [k']}[|S \cap J| = 1] \cdot \frac{1}{|S|} \\
&= \sum_{S: S \ni i} \frac{\widehat{f}(S)^2}{|S|} \cdot \sum_{m=0}^{\log(10k)} \mathop{\mathbf{Pr}}_{J \subseteq_{2^{-m}} [k']}[|S \cap J| = 1]
\end{aligned}$$

It therefore suffices to show that for any non-empty set S such that $|S| \leq k$ it holds that

$$\frac{1}{2} \leq \sum_{m=0}^{\log(10k)} \Pr_{J^{(m)} \subseteq_{2^{-m}} [k']} [|S \cap J^{(m)}| = 1] \leq 2 \, . \tag{2.3}$$

From which it is clear that $\lambda_i \leq 2 \cdot \sum_{S:S\ni i} \frac{\widehat{f}(S)^2}{|S|} = 2 \cdot \mathbf{NInf}_i[f]$ and similarly $\lambda_i \geq \frac{1}{2} \sum_{\substack{S\ni i,\\ |S|\leq k}} \frac{\widehat{f}(S)^2}{|S|} = \frac{1}{2}\mathbf{NInf}_i^{\leq k}[f]$.

We move to prove Equation (2.3). The first observation is that an equivalent way to sample $J^{(m)} \subseteq_{2^{-m}} [k']$ is to sample m independent set $J_1^{(m)}, \dots, J_m^{(m)} \subseteq_{1/2} [k']$ and take their intersection $J^{(m)} = J_1^{(m)} \cap \dots \cap J_m^{(m)}$. Furthermore, by linearity of expectation

$$\sum_{m=0}^{\infty} \Pr_{J^{(m)} \subseteq_{2^{-m}} [k']} [|S \cap J| = 1] = \sum_{m=0}^{\infty} \mathop{\mathbf{E}}_{\substack{J_1^{(m)} \subseteq_{1/2} [k'],\\ J_2^{(m)} \subseteq_{1/2} [k'],\\ \dots}} \left[\mathbb{1}_{|S\cap J_1^{(m)} \cap \dots \cap J_m^{(m)}|=1}\right] = \mathop{\mathbf{E}}_{\substack{J_1 \subseteq_{1/2} [k'],\\ J_2 \subseteq_{1/2} [k'],\\ \dots}} \left[\sum_{m=0}^{\infty} \mathbb{1}_{|S\cap J_1 \cap \dots \cap J_m|=1}\right]$$

which in essence means that the choices for $J_1^{(1)}, J_1^{(2)}, \dots$ can be the same set J_1, and similarly for any J_i.

To analyze the latter expectation, we note that it can be described as the expected value of the following random process:

```
1 $X \leftarrow 0$
2 for $i = 1, 2, \dots, \log(10k)$ do
3     if $S = \emptyset$ then
4         halt!;
5     if $|S| = 1$ then
6         increment $X$;
7     Sample $J_i \subseteq_{1/2} [k']$;
8     $S \leftarrow S \cap J_i$;
```

It therefore suffices to show that the expected value of the above random process is bounded in $[1/2, 2]$. In the analysis, we consider also the infinite horizon process that keeps on going until $S = \emptyset$. We observe that the expected values of both processes depend only on the size of the initial S from symmetry. For any $t \in \{0, 1, \dots, k'\}$, denote by F_t the expected value of the infinite horizon process starting with a set S of size t. For the finite horizon process with i iterations, we let the expected value be denoted by $F_t^{(i)}$. We observe that $F_0 = 0$, and furthermore that $F_1 = 2$ since starting from a set of size 1 the random variable X would behave like geometric random variable with $p = 1/2$. Similarly, $F_1^{(i)} = 2 - \frac{1}{2^{i-1}}$ as it is the minimum of i and a geometric random variable with $p = 1/2$.

Furthermore, for the infinite horizon process, we observe that we have the following recurrence

$$F_t = \sum_{a=0}^{t} \frac{\binom{t}{a}}{2^t} \cdot F_a,$$

for $t \geq 2$ or equivalently

$$F_t \cdot (1 - 2^{-t}) = \sum_{a=0}^{t-1} \frac{\binom{t}{a}}{2^t} \cdot F_a.$$

We show by induction that $1/2 < F_t^{(\log 10k)} \le 2$ for $t \ge 1$. The base case $t = 1$ was discussed above. Applying the induction hypothesis we have

$$F_t \cdot (1-2^{-t}) = \sum_{a=0}^{t-1} \frac{\binom{t}{a}}{2^t} \cdot F_a \le \sum_{a=0}^{t-1} \frac{\binom{t}{a}}{2^t} \cdot 2 \le (1-2^{-t}) \cdot 2.$$

Dividing both sides by $(1-2^{-t})$ gives the inequality $F_t \le 2$, which implies that $F_t^{(\log 10k)} \le 2$.

For the lower bound, we consider the indicator random variable $Y_t^{(i)}$, where $t = |S|$, which equals 1 if $|S| = 1$ at some point during the above process before iteration i. We note that $Y_t^{(\log 10k)}$ is a lower bound for the value of X in the finite horizon process, and Y_t is a lower bound for the value of X at the end of the infinite horizon process. First, we claim that $\mathbf{E}[Y_t] = \mathbf{Pr}[Y_t = 1] \ge 2/3$ for all $t \ge 1$. The base case of $t = 1$ is certainly true, and we also have, similar to before, that

$$\begin{aligned}\mathbf{E}[Y_t] \cdot (1-2^{-t}) &= \sum_{a=0}^{t-1} \frac{\binom{t}{a}}{2^t} \mathbf{E}[Y_a] \\ &\ge 0 \cdot \frac{1}{2^t} + 1 \cdot \frac{t}{2^t} + \frac{2}{3} \cdot \underbrace{\sum_{a=2}^{t-1} \frac{\binom{t}{a}}{2^t}}_{1-\frac{2+t}{2^t}} \\ &= \frac{2}{3} + \frac{t - \frac{2}{3}(2+t)}{2^t} \ge \frac{2}{3} + \frac{t/3 - 4/3}{2^t} \ge \frac{2}{3} - \frac{2/3}{2^t} = \frac{2}{3} \cdot (1-2^{-t})\end{aligned}$$

which holds for all $t \ge 2$, and thus $\mathbf{Pr}[Y_t = 1] \ge 2/3$. However, this only holds for the infinite horizon random process. Let A be the event that $S = \emptyset$ by iteration $\log 10k$, and note that $\mathbf{Pr}[A] = \mathbf{Pr}[\mathrm{Bin}(|S|, \frac{1}{10k}) = 0] \ge \mathbf{Pr}[\mathrm{Bin}(k, \frac{1}{10k}) = 0] = \left(1 - \frac{1}{10k}\right)^k \ge 1 - \frac{k}{10k} = 0.9$. Finally, we claim that for all $t \ge 2$ we have that $\mathbf{Pr}[Y_t^{(\log 10t)}] \ge 1/2$. Note that for Y_t to happen, it must be the case that either $\overline{A}$ happens or $Y_t^{(\log 10t)}$ happens. Thus, by a union bound

$$\tfrac{2}{3} \le \mathbf{Pr}[Y_t = 1] \le \mathbf{Pr}[Y_t^{(\log 10t)} = 1] + \mathbf{Pr}[\overline{A}] \le \mathbf{Pr}[Y_t^{(\log 10t)} = 1] + 0.1\ ,$$

which implies $\mathbf{Pr}[Y_t^{(\log 10t)} = 1] > 1/2$. Finally, $F_t^{(\log 10t)} \ge \mathbf{Pr}[Y_t^{(\log 10t)} = 1] > 1/2$ as desired. □

2.3 A $2^{\widetilde{O}(\sqrt{k})}$-query Tolerant Junta Tester

In this section, we prove Theorem 1.8.1. Throughout this section, we assume that we already applied Algorithm 7 to reduce the number of coordinate oracles to $O(k/\varepsilon^2)$. We denote by $\mathcal{D}$ the set of oracles we get, and by $\mathcal{S} \subseteq [n]$ the set of coordinate to which they are oracles to. Suppose that the best k-junta approximation of f is a junta-on-T, for a set $T \subseteq \mathcal{S}$ of size k. We call T the "target set". Note that T is unknown to the algorithm, and in fact, identifying T (or a close approximation to T) from all subsets of size k of $\mathcal{S}$ is the crux of the problem.

We start with the observation that if we were somehow able to identify all of the variables of T that capture most of the Fourier mass above level thr, then we could simply restrict

f by randomly fixing these variables, leaving us with the task of identifying the best k-junta approximation of f, given that we know the best k-junta has most its Fourier mass below level thr. For the latter case, there are only $\binom{|\mathcal{S}|}{\mathsf{thr}}$ Fourier coefficients to estimate, and estimating these to sufficient accuracy allows one to estimate the the correlation f has with any subset $U \subseteq \mathcal{S}$ such that $|U| \le k$.

We are now ready to present the details of the algorithm. The algorithm can be broken down into two main steps. First, we find, with high probability, a set $B \subseteq T$ that captures almost all Fourier mass of T above level thr. This first step, which we call "phase one", closely resembles the techniques in Section 2.2 in that we utilize a series of random restrictions to estimate normalized influences. The main difference is that rather than considering normalized influences of individual coordinates, we now consider normalized influences of sets of size thr. The goal of phase one is to produce at least one subset B of our target set T which effectively captures most of the Fourier mass within T above level thr. Once we have done that, we have reduced to the scenario of the closest k-junta to f having most of its Fourier mass below level thr, which can be solved via estimating all of the Fourier coefficients below level thr.

2.3.1 Phase One: The Higher Levels

First, we prove an analogous theorem to Theorem 2.2.11, which relates $\lambda_U[f]$ to $\mathbf{NInf}_U[f]$ for all U:

Theorem 2.3.1. *Let $f : \{\pm 1\}^\ell \to \mathbb{R}$. Let $U \subseteq [\ell]$, where $\ell = |\mathcal{D}|$ and $|U| \le k$. Let*

$$\lambda_U[f] = \sum_{m=0}^{2|U|\log(10k)} \lambda_U^{\approx p^{-m}}[f], \qquad \textit{where} \qquad \lambda_U^{\approx p^{-m}}[f] = \mathop{\mathbf{E}}_{(J,z)\sim\mathcal{R}_{p^m}}[\widehat{f_{\bar{J}\to z}}(U)^2]$$

for $p = 1 - \frac{1}{2|U|}$. Then, $\frac{1}{2} \cdot \mathbf{NInf}_U^{\le k}[f] \le \lambda_U[f] \le 3 \cdot \mathbf{NInf}_U[f]$.

Again, we postpone the proof of this to the end of this section in Section 2.3.3.

The definition of $\lambda_U[f]$ is naturally algorithmic, and therefore we can design the following algorithm to approximate the values of $\lambda_U[f]$ for all sets U of size $\mathsf{thr} = \sqrt{\varepsilon k}$.

Algorithm 8: Estimating λ_U's

```
Input: $f : \{\pm 1\}^{k'} \to [-1, 1]$ along with a randomized algorithm A computing $f$
       (recall Def. 1.9.9). Parameters $1 - \delta$ (confidence), $\varepsilon$ (additive error) and $k$.
Output: Estimates $\{\widetilde{\lambda}_U\}_{|U|=\mathsf{thr}}$ for $\{\lambda_U\}_{|U|=\mathsf{thr}}$.
1  Let $m = \mathsf{poly}(k, k', 1/\varepsilon, \log(1/\delta))$
2  Initialize $\widetilde{\lambda}_U = 0$ for all $U \subseteq [k']$, $|U| = \mathsf{thr} = \sqrt{\varepsilon k}$
3  Let $p = \left(1 - \frac{1}{2\mathsf{thr}}\right)$
4  for $d = 0$ to $2\mathsf{thr} \log 10k$ do
5      Initialize $\widetilde{\lambda}_U^{\approx p^{-d}} = 0$ for all $U \subseteq [k']$ such that $|U| = \mathsf{thr}$
6      repeat $m$ times
7          Let $(J, z) \sim \mathcal{R}_{p^d}$ be a $p^d$-random restriction.
8          Estimate $\widehat{f_{\bar{J} \to z}}(U)$ for all $U \subseteq J$ of size $\mathsf{thr}$ up to additive error $\frac{\varepsilon}{12\mathsf{thr} \log(10k)}$
            with probability $1 - \frac{\delta}{\binom{k'}{\mathsf{thr}} m \cdot 2\mathsf{thr} \log(10k)}$ using Claim 1.9.10 and algorithm A.
            Denote by $\widetilde{f_{\bar{J} \to z}}(U)$ the estimated Fourier coefficient.
9          Update $\widetilde{\lambda}_U^{\approx p^{-d}} = \widetilde{\lambda}_U^{\approx p^{-d}} + \widetilde{f_{\bar{J} \to z}}(U)^2$ for all $U \subseteq J$ of size $\mathsf{thr}$.
10     Let $\widetilde{\lambda}_U^{\approx p^{-d}} = \widetilde{\lambda}_U^{\approx p^{-d}} / m$ for all $U \subseteq J$ of size $\mathsf{thr}$;
11 Let $\widetilde{\lambda}_U = \sum_d \widetilde{\lambda}_U^{\approx p^{-d}}$;
12 return $\{\widetilde{\lambda}_U\}_{|U|=\mathsf{thr}}$
```

Lemma 2.3.2. *With probability at least $1 - \delta$ we have that for all $U \subseteq [k']$ of size thr it holds that $|\widetilde{\lambda}_U - \lambda_U[f]| \le \varepsilon$.*

Proof. This proof closely follows that of Lemma 2.2.12. If $U \not\subseteq J$ the Fourier coefficient of $\widehat{f_{\bar{J} \to z}}(U)$ is 0 and so our estimate is correct in that case. In the case $U \subseteq J$, each estimation of the Fourier coefficient is correct up to additive error $\eta = \frac{\varepsilon}{12\mathsf{thr} \log(10k)}$ with probability at least $1 - \delta / \exp(k, k', m)$. Thus, we get that $\widetilde{f_{\bar{J} \to z}}(U)^2 = (\widehat{f_{\bar{J} \to z}}(U) \pm \eta)^2 = \widehat{f_{\bar{J} \to z}}(U)^2 \pm 2\eta |\widehat{f_{\bar{J} \to z}}(U)| \pm \eta^2 = \widehat{f_{\bar{J} \to z}}(U)^2 \pm 3\eta$. Furthermore, we have that $\mathbf{E}_{(J,z) \sim \mathcal{R}_{p^d}}[\widehat{f_{\bar{J} \to z}}(U)^2] = \lambda_U^{\approx p^{-d}}$, thus by Fact 1.9.1 we have that the empirical mean of $m = \mathsf{poly}(1/\varepsilon, \mathsf{poly}(k), \mathsf{poly}(k'), \log(1/\delta))$ copies of $\widetilde{f_{\bar{J} \to z}}(U)^2$ is within additive error $\varepsilon / (4\mathsf{thr} \log(10k))$ from $\lambda_U^{\approx p^{-d}}$ with probability at least $1 - \frac{\delta}{\binom{k'}{\mathsf{thr}} m \cdot 2\mathsf{thr} \log(10k)}$. By union bound, all these estimates are within the error bound, and we get that

$$\left| \widetilde{\lambda}_U^{\approx p^{-d}} - \lambda_U^{\approx p^{-d}} \right| \le 3\eta + \varepsilon / (4\mathsf{thr} \log(10k)) \le \varepsilon / (2\mathsf{thr} \log(10k)).$$

Overall, we get that $|\widetilde{\lambda}_U - \lambda_U[f]| \le \varepsilon$ for all $|U| = \mathsf{thr}$ with probability at least $1 - \delta$. □

Since we are sampling sets of size thr, we need to sample at most $k/\mathsf{thr} = \sqrt{k/\varepsilon} =: \alpha$ distinct subsets of T of size thr in order to capture all the potential mass of T above level thr.

Algorithm 9: Branching Process

```
Input: $f$ (target function), $\mathcal{D}$ (where $\mathcal{D}$ are coordinate oracles for $\mathcal{S}$) a current
       depth $t$, a current subset $\mathcal{D}' \subseteq \mathcal{D}$ of coordinate oracles, $\varepsilon$, $\delta$
Output: Return collection of subsets of $\mathcal{D}$ of size at most $k$.
1  Let $\alpha = k/\mathsf{thr} = \sqrt{k/\varepsilon}$
2  Let $r = O(1/\varepsilon^2)$ and $\ell = 2(r+1)^{3\alpha+\log(2/\delta)}$
   /* $r+1$ is the branching factor, and $\ell$ is an upper bound on the number
      of nodes in the branching process (the process depth is
      $3\alpha+\log(2/\delta)$).                                              */
3  if $t = 3\alpha+\log(2/\delta)$ or $|\mathcal{D}'| > k - \mathsf{thr}$ then
4      return $\{\mathcal{D}'\}$
5  Let $\{g_1, ..., g_{k'}\} = \mathcal{D} - \mathcal{D}'$ and $\{g_{k'+1}, ..., g_{|\mathcal{D}|}\}$ where $k' = |\mathcal{D}| - |\mathcal{D}'|$
6  Sample $z \in \{\pm 1\}^{|\mathcal{D}'|}$. Let $f' : \{\pm 1\}^{k'} \to \mathbb{R}$ be the function defined by

     $$f'(x_1, \dots, x_{k'}) = \mathop{\mathbf{E}}_{y\sim\{\pm 1\}^n}[f(y) | g_1(y) = x_1, \dots, g_{k'}(y) = x_{k'}, g_{k'+1}(y) = z_1, \dots, g_{|\mathcal{D}|}(y) = z_{|\mathcal{D}'|}],$$

   and let A be the randomized algorithm for $f'$ from Theorem 2.2.10.
7  Apply Algorithm 8 on $f'$ using the randomized algorithm A for $f'$ with confidence
   $1 - \frac{\delta}{2\ell}$ and accuracy $\frac{\varepsilon^2}{48\cdot\binom{|\mathcal{D}|}{\mathsf{thr}}} \implies \widetilde{\lambda} = \{\widetilde{\lambda}_U\}_{|U|=\mathsf{thr}}$.
8  Let our distribution $P$ be defined by $\widetilde{\lambda}$, normalized appropriately
9  Sample $M_1, ..., M_r \sim \widetilde{\lambda}$
10 Let $\mathcal{L} = \{\}$.
11 for $M = \emptyset, M_1, ..., M_r$ do
12     $\mathcal{L} = \mathcal{L} \cup \mathrm{BranchingProcess}(f, \mathcal{D}, t+1, \mathcal{D}' \cup \{g_i : i \in M\}, \varepsilon, \delta)$
13 return $\mathcal{L}$
```

Lemma 2.3.3. *With probability at least $1-\delta$, at least one of the subsets Algorithm 9 returns is a set of coordinate oracles to $B \subseteq T$ such that*

$$\mathop{\mathbf{E}}_{z}\left[\sum_{\substack{S\subseteq T\setminus B \\ |S|>\mathsf{thr}}} \widehat{f_{B\to z}}(S)^2\right] \le \varepsilon^2/4. \tag{2.4}$$

The reason for Equation (2.4) becomes clear in Section 2.3.2, where we show that assuming the inequality, we lose at most an additive error of $\varepsilon/2$ to the nearest k-junta if we ignore the Fourier mass above level thr after restricting B. As before, in order to prove the above lemma, we prove a claim capturing the algorithm's progress towards satisfying Equation (2.4).

We denote the event $\mathcal{E}$ that in the entire execution of Algorithm 9 all of the $\widetilde{\lambda}_U$ were $\varepsilon^2/48 \cdot \binom{|\mathcal{D}|}{\mathsf{thr}}$ close to the real λ_U. We note that by a union bound, this happens with probability at least $1-\delta/2$.

Suppose again that T is the (unknown) set of k coordinates for which the best k-junta approximating f is a junta on T. If T has Fourier mass less than $\varepsilon^2/4$ above level thr then one of the subsets that Algorithm 9 will return is the empty set, which satisfies the claim. Therefore, henceforth we assume that T has at least $\varepsilon^2/4$ Fourier mass above level thr. We show that in such a case, each M_i for $i = 1, \ldots, r$ will be a subset of T with probability at least $\Omega(\varepsilon^2)$.

Claim 2.3.4. *Assume $\mathcal{D}'$ are coordinate oracles to $\mathcal{S}' \subseteq T$. Suppose also that*

$$\mathop{\mathbf{E}}_{z}\left[\sum_{\substack{S\subseteq T\setminus\mathcal{S}'\\|S|>\mathsf{thr}}} \widehat{f_{\mathcal{S}'\to z}}(S)^2\right] > \varepsilon^2/4.$$

Then, conditioned on $\mathcal{E}$, when running the Branching Process on $\mathcal{D}'$, each M_i will be with probability at least $\varepsilon^2/40$ a collection of thr *new coordinate oracles to coordinates in T.*

Proof. Similar to the proof of Claim 2.2.14, denote by $f_z = (f_{\mathsf{avg},\mathcal{S}})_{\mathcal{S}'\to z}$, and note that f' is up to relabeling of coordinates the same function as f_z. Denote $V \subseteq T$ as the part of the target set we have not yet sampled, so $V = T \setminus \mathcal{S}'$. Then, using our assumption, we have that

$$\begin{aligned}
\varepsilon^2/4 &< \mathop{\mathbf{E}}_{z}\left[\sum_{\substack{S\subseteq V\\|S|>\mathsf{thr}}} \widehat{f_{\mathcal{S}'\to z}}(S)^2\right] \\
&= \sum_{\substack{S\subseteq V\\|S|>\mathsf{thr}}} \sum_{\substack{R\subseteq[n]:\\R\cap\overline{\mathcal{S}'}=S}} \widehat{f}(R)^2 && \text{(Fact 1.9.7)}\\
&= \sum_{\substack{S\subseteq V\\|S|>\mathsf{thr}}} \sum_{\substack{R\subseteq\mathcal{S}:\\R\cap\overline{\mathcal{S}'}=S}} \widehat{f}(R)^2 && (\text{if } R\not\subseteq\mathcal{S} \text{ then } R\cap\overline{\mathcal{S}'}\neq S)\\
&= \sum_{\substack{S\subseteq V\\|S|>\mathsf{thr}}} \sum_{\substack{R\subseteq\mathcal{S}:\\R\cap\overline{\mathcal{S}'}=S}} \widehat{f_{\mathsf{avg},\mathcal{S}}}(R)^2\\
&= \mathop{\mathbf{E}}_{z}\left[\sum_{\substack{S\subseteq V\\|S|>\mathsf{thr}}} \widehat{f_z}(S)^2\right].
\end{aligned}$$

Next, by applying Theorem 2.3.1, we have that

$$\sum_{\substack{U\subseteq V\\|U|=\mathsf{thr}}} \lambda_U[f_z] \geq \frac{1}{2}\sum_{U\subseteq V} \mathbf{NInf}_U^{\leq k}[f_z] \geq \frac{1}{2}\sum_{U\subseteq V:|U|=\mathsf{thr}} \sum_{S:U\subseteq S\subseteq V} \frac{\widehat{f_z}(S)^2}{\binom{|S|}{|U|}} = \frac{1}{2}\sum_{\substack{|S|>\mathsf{thr}\\S\subseteq V}} \widehat{f_z}(S)^2$$

Then, using the assumption that $\mathcal{E}$ happens, the $\widetilde{\lambda_U}$ are $\frac{\varepsilon^2}{48\cdot\binom{|S|}{\mathsf{thr}}}$-accurate, and we get that

$$\sum_{\substack{U\subseteq V\\|U|=\mathsf{thr}}} \widetilde{\lambda_U}[f_z] \geq \frac{1}{2}\sum_{\substack{|S|>\mathsf{thr}\\S\subseteq V}} \widehat{f_z}(S)^2 - \frac{\varepsilon^2}{48\cdot\binom{|S|}{\mathsf{thr}}}\cdot\binom{k}{\mathsf{thr}} \geq \frac{1}{2}\sum_{\substack{|S|>\mathsf{thr}\\S\subseteq V}} \widehat{f_z}(S)^2 - \frac{\varepsilon^2}{48}.$$

On the other hand, again by applying Theorem 2.3.1, we have that

$$\sum_{\substack{U\subseteq\mathcal{S}\\|U|=\mathsf{thr}}} \lambda_U[f_z] \leq 3\sum_{\substack{U\subseteq\mathcal{S}\\|U|=\mathsf{thr}}} \mathbf{NInf}_U[f_z] \leq 3\mathbf{W}^{\geq\mathsf{thr}}[f_z] \leq 3.$$

This implies that $\sum_U \widetilde{\lambda_U} \leq 3 + \frac{\varepsilon^2}{48 \cdot \binom{|S|}{\mathsf{thr}}} \cdot \binom{|S|}{\mathsf{thr}} \leq 4$. Overall, the probability to sample $U \subseteq V$ is at least

$$\frac{1}{4}\left(\frac{1}{2}\sum_{\substack{|S|>\mathsf{thr}\\ S\subseteq V}} \widehat{f}_z(S)^2 - \frac{\varepsilon^2}{48}\right) = \frac{1}{8}\sum_{\substack{|S|>\mathsf{thr}\\ S\subseteq V}} \widehat{f}_z(S)^2 - \frac{\varepsilon^2}{4\cdot 48}.$$

Taking an expectation over z, we see that the probability to sample a subset of V is at least

$$\mathop{\mathbf{E}}_{z}\left[\frac{1}{8}\sum_{\substack{|S|>\mathsf{thr}\\ S\subseteq V}} \widehat{f}_z(S)^2 - \frac{\varepsilon^2}{4\cdot 48}\right] \geq \frac{1}{8}\cdot\frac{\varepsilon^2}{4} - \frac{\varepsilon^2}{4\cdot 48} \geq \frac{\varepsilon^2}{40}.$$ □

We are now ready to prove Lemma 2.3.3.

Proof of Lemma 2.3.3. By Claim 2.3.4, if our special set T has at least $\varepsilon^2/4$ mass on the levels above thr, then if we sample according to our distribution $\widetilde{\lambda} = \{\widetilde{\lambda}_U\}_{|U|=\mathsf{thr}}$, we will see $U \subseteq T$ with probability at least $\varepsilon^2/40$. Then, if we sample $r = O(\varepsilon^{-2})$ subsets in Algorithm 9, applying the multiplicative Chernoff bound in Fact 1.9.1, we see at least one subset of T with probability at least $p \geq 0.9$ each time we sample $M_1, ..., M_r$ in Algorithm 9. In order for Algorithm 9 to successfully find B_i with the desired property, it suffices to have sampled from T at least α times in our branching process. Therefore, we can treat our $N := (3\alpha + \log(2/\delta))$ depth branching process as a $X = \mathrm{Bin}(N, p)$ random variable. Applying a standard Chernoff bound (second case in Fact 1.9.1), we have that our probability of failure is

$$\begin{aligned}
\mathbf{Pr}[X < \alpha] &= \mathbf{Pr}[\overline{X} < \tfrac{\alpha}{N}] \\
&= \mathbf{Pr}[\overline{X} < 0.9 - (0.9 - \tfrac{\alpha}{N})] \\
&\leq \exp(-2N(0.9 - \tfrac{\alpha}{N})^2) && \text{(Using Fact 1.9.1)} \\
&\leq \exp(-2N(0.81 - 2\tfrac{\alpha}{N})) \\
&\leq \exp(-1.5N + 4\alpha) \\
&\leq \exp(-\log(2/\delta)) = \delta/2.
\end{aligned}$$

This shows that, by a union bound with event $\mathcal{E}$, one of the branches of our algorithm find's a B_i satisfying Equation (2.4) with probability at least $1 - \delta$. □

Claim 2.3.5. *The query complexity of phase one of the algorithm for constant δ (failure probability) is $2^{\widetilde{O}(\sqrt{k/\varepsilon})}$.*

Proof. All of our queries to f in phase one come from estimating fourier coefficients using Claim 1.9.10 in Algorithm 8. We require that the estimated Fourier coefficients be accurate to within $1/\mathsf{poly}(k, 1/\varepsilon)$ with confidence $1 - O(1/\ell) = 1 - 2^{-\widetilde{\Omega}(\sqrt{k/\varepsilon})}$, which is possible via Fact 1.9.1 with query complexity $\mathsf{poly}(k/\varepsilon)$. However, we do this $O(\ell) = 2^{\widetilde{O}(\sqrt{k/\varepsilon})}$ times during the branching process, which yields the final overall query complexity. □

2.3.2 Phase Two: The Lower Levels

Now, we are ready to use Algorithm 9. Our strategy will be to take the subsets outputted from Algorithm 9 one at time, randomly fixing those coordinates, and then treating this restricted version of f as if all its Fourier mass were below level thr (recall that $\mathsf{thr} = \sqrt{\varepsilon k}$). Let T be the target set of size k on which there exists a k-junta which best approximates f. Assume that the first part of the algorithm is successful in yielding at least one $B \subseteq T$ such that:

$$\mathop{\mathbf{E}}_{z\in\{\pm1\}^B}\Big[\sum_{\substack{S\subseteq T\setminus B\\ |S|>\mathsf{thr}}} \widehat{f_{B\to z}}(S)^2\Big] \le \varepsilon^2/4. \tag{2.5}$$

Let g be the maximizer of $\max_{g'\in\mathcal{J}_T}\mathbf{E}[fg']$. Recall that by Claim 1.9.2 we have that $g = \mathsf{sgn}(f_{\mathsf{avg},T})$ and

$$\begin{aligned}\mathsf{corr}(f,\mathcal{J}_T) = \mathbf{E}[fg] = \mathop{\mathbf{E}}_{y\in\{\pm1\}^T}[|f_{\mathsf{avg},T}(y)|] &= \mathop{\mathbf{E}}_{z\in\{\pm1\}^B}\ \mathop{\mathbf{E}}_{x\in\{\pm1\}^{T\setminus B}}\Big|(f_{\mathsf{avg},T})_{B\to z}(x)\Big| &(2.6)\\ &= \mathop{\mathbf{E}}_{z\in\{\pm1\}^B}\ \mathop{\mathbf{E}}_{x\in\{\pm1\}^{T\setminus B}}\Big|\sum_{S\subseteq T\setminus B}\widehat{f_{B\to z}}(S)\chi_S(x)\Big| &(2.7)\end{aligned}$$

Furthermore, using the assumption in Eq. (2.5) it is an easy calculation to show that (2.7) equals

$$\mathop{\mathbf{E}}_{z\in\{\pm1\}^B}\ \mathop{\mathbf{E}}_{x\in\{\pm1\}^{T\setminus B}}\Big|\sum_{S\subseteq T\setminus B, |S|\le\mathsf{thr}}\widehat{f_{B\to z}}(S)\chi_S(x)\Big| \pm \varepsilon/2.$$

Similarly, for any set $U \subseteq \mathcal{S}$ of size k containing B (think of U as a candidate for T) we have that the best correlation between a junta-on-U and f is

$$\mathsf{corr}(f,\mathcal{J}_U) = \mathop{\mathbf{E}}_{z\in\{\pm1\}^B}\ \mathop{\mathbf{E}}_{x\in\{\pm1\}^{U\setminus B}}\Big|\sum_{S\subseteq U\setminus B}\widehat{f_{B\to z}}(S)\chi_S(x)\Big|. \tag{2.8}$$

Now, however, the right hand side in Eq. (2.8) is not necessarily approximated by the low-degree counterpart as above for T. Indeed, we would like to estimate Eq. (2.8) for all candidates $U \subseteq \mathcal{S}$ of size k containing B, and pick the set with best estimated correlation. Based on our assumption on T, we can replace $\sum_{S\subseteq U\setminus B}\widehat{f_{B\to z}}(S)\chi_S(x)$ with its low-degree part $\sum_{S\subseteq U\setminus B,|S|\le\mathsf{thr}}\widehat{f_{B\to z}}(S)\chi_S(x)$ for $U = T$, but its not clear whether we can do it in general.

In particular, if U satisfies

$$\mathop{\mathbf{E}}_{z\in\{\pm1\}^B}\Big[\sum_{\substack{S\subseteq U\setminus B,\\ |S|>\mathsf{thr}}} \widehat{f_{B\to z}}(S)^2\Big] > \varepsilon^2/4, \tag{2.9}$$

then taking the low-degree part can give an overestimate to the correlation with the best junta on U.[1] We settle for an estimate that is ε-accurate for the target set T assuming it satisfies Equation (2.5), and is not overestimating by more than ε for any other set $U \supseteq B$

[1]To see a simple example of how this can happen, consider $f(x,y) = 1-x-y+xy$. Then one can verify that $\mathbf{E}[|f(x,y)|] = 1 < 1.5 = \mathbf{E}[|1-x-y|]$.

of size k. Towards this goal, we first apply a noise operator that would essentially eliminate most of the contribution from sets larger than $\sqrt{k/\varepsilon}\log(1/\varepsilon)$ regardless of whether U satisfies Eq. (2.20) or not. This is captured by the following claim.

Claim 2.3.6. *Let $\rho = 1 - \sqrt{\varepsilon/k}$, $z \in \{\pm1\}^B$ and denote by $h = f_{B\to z}$ and $h^{\mathsf{low}} = h^{\leq(\sqrt{k/\varepsilon})\cdot\log(1/\varepsilon)}$ (i.e., h^{low} is the truncated Fourier expansion of h that zeroes out all Fourier coefficients above level $(\sqrt{k/\varepsilon})\cdot\log(1/\varepsilon)$). For any $U : B \subseteq U \subseteq \mathcal{S}$ it holds that*

$$\left|\mathsf{corr}\left(T_\rho h, \mathcal{J}_U\right) - \mathsf{corr}\left(T_\rho h^{\mathsf{low}}, \mathcal{J}_U\right)\right| \leq \varepsilon.$$

Proof. We have

$$\begin{aligned}
&\left|\mathsf{corr}\left(T_\rho h, \mathcal{J}_U\right) - \mathsf{corr}\left(T_\rho h^{\mathsf{low}}, \mathcal{J}_U\right)\right| \\
&= \left|\mathop{\mathbf{E}}_{x\in\{\pm1\}^{U\setminus B}}\left|\sum_{S\subseteq U\setminus B}\widehat{h}(S)\chi_S(x)\rho^S\right| - \mathop{\mathbf{E}}_{x\in\{\pm1\}^{U\setminus B}}\left|\sum_{\substack{S\subseteq U\setminus B,\\ |S|\leq(\sqrt{k/\varepsilon})\cdot\log(1/\varepsilon)}}\widehat{h}(S)\chi_S(x)\rho^S\right|\right| \\
&\leq \mathop{\mathbf{E}}_{x\in\{\pm1\}^{U\setminus B}}\left|\sum_{\substack{S\subseteq U\setminus B,\\ |S|>(\sqrt{k/\varepsilon})\cdot\log(1/\varepsilon)}}\widehat{h}(S)\chi_S(x)\rho^{|S|}\right| \\
&\leq \sqrt{\mathop{\mathbf{E}}_{x\in\{\pm1\}^{U\setminus B}}\left(\sum_{\substack{S\subseteq U\setminus B,\\ |S|>(\sqrt{k/\varepsilon})\cdot\log(1/\varepsilon)}}\widehat{h}(S)\chi_S(x)\rho^{|S|}\right)^2} \\
&= \sqrt{\sum_{\substack{S\subseteq U\setminus B,\\ |S|>(\sqrt{k/\varepsilon})\cdot\log(1/\varepsilon)}}\widehat{h}(S)^2\rho^{2|S|}} \leq \sqrt{\rho^{2(\sqrt{k/\varepsilon})\cdot\log(1/\varepsilon)}} \leq \varepsilon.
\end{aligned}$$

□

Next, we show that applying a noise operator to f does not affect its correlation with a set U of size k, under the condition that most of the Fourier mass of $f_{B\to z}$ falls on the lower levels, i.e., $\mathbf{E}_z\left[\sum_{S\subseteq U\setminus B,|S|\geq\sqrt{k}}\widehat{f_{B\to z}}(S)^2\right] \leq \varepsilon^2/4$. Recall that this is what was guaranteed with high probability from the output of Algorithm 9 for our target set T.

Claim 2.3.7. *Let $\rho = 1-\sqrt{k/\varepsilon}$. Given $U : B \subseteq U \subseteq \mathcal{S}$ such that $\mathbf{E}_z\left[\sum_{\substack{S\subseteq U\setminus B,\\ |S|\geq\mathsf{thr}}}\widehat{f_{B\to z}}(S)^2\right] \leq \varepsilon^2/4$, we have that*

$$\left|\mathop{\mathbf{E}}_z\mathsf{corr}(T_\rho(f_{B\to z}), \mathcal{J}_U) - \mathop{\mathbf{E}}_z\mathsf{corr}(f_{B\to z}, \mathcal{J}_U)\right| \leq 1.2\varepsilon.$$

Proof. Similar to the proof of Claim 2.3.6, we have

$$\begin{aligned}
&\left| \mathop{\mathbf{E}}_{\substack{z\in\{\pm1\}^B\\ x\in\{\pm1\}^{U\setminus B}}} \left| \sum_{S\subseteq U\setminus B} \widehat{f_{B\to z}}(S)\chi_S(x)\right| - \mathop{\mathbf{E}}_{\substack{z\in\{\pm1\}^B\\ x\in\{\pm1\}^{U\setminus B}}} \left| \sum_{S\subseteq U\setminus B} \widehat{f_{B\to z}}(S)\chi_S(x)\cdot\rho^{|S|}\right| \right| \\
&\le \mathop{\mathbf{E}}_{\substack{z\in\{\pm1\}^B\\ x\in\{\pm1\}^{U\setminus B}}} \left| \sum_{S\subseteq U\setminus B} \widehat{f_{B\to z}}(S)\chi_S(x)(1-\rho^{|S|})\right| \\
&\le \sqrt{\mathop{\mathbf{E}}_{\substack{z\in\{\pm1\}^B\\ x\in\{\pm1\}^{U\setminus B}}} \left(\sum_{S\subseteq U\setminus B} \widehat{f_{B\to z}}(S)\chi_S(x)(1-\rho^{|S|})\right)^2} \\
&= \sqrt{\mathop{\mathbf{E}}_{z\in\{\pm1\}^B} \left[\sum_{S\subseteq U\setminus B} \widehat{f_{B\to z}}(S)^2\cdot(1-\rho^{|S|})^2\right]} \\
&\le \sqrt{\mathop{\mathbf{E}}_{z\in\{\pm1\}^B} \left[\sum_{S\subseteq U\setminus B:|S|\le \mathsf{thr}} \widehat{f_{B\to z}}(S)^2\cdot(1-\rho^{|S|})^2 + \sum_{S\subseteq U\setminus B:|S|> \mathsf{thr}} \widehat{f_{B\to z}}(S)^2\cdot(1-\rho^{|S|})^2\right]} \\
&\le \sqrt{(1-\rho^{\mathsf{thr}})^2+\varepsilon^2/4} \le \sqrt{\varepsilon^2+\varepsilon^2/4} \le 1.2\cdot\varepsilon. \qquad \square
\end{aligned}$$

The next lemma gives an algorithm that on any B, satisfying Equation (2.5), outputs $U : B\subseteq U\subseteq \mathcal{S}$ with $\mathsf{corr}(f,\mathcal{J}_U)\ge \mathsf{corr}(f,\mathcal{J}_T)-O(\varepsilon)$, with high probability.

Lemma 2.3.8 (Algorithm and Analysis for Phase-Two)**.** *Let $\varepsilon,\delta>0$. There's an algorithm that with probability at least $1-\delta$, gives ε-accurate estimates $\widetilde{c_U}$ to*

$$c_U = \mathop{\mathbf{E}}_{z\in\{\pm1\}^B} \mathop{\mathbf{E}}_{x\in\{\pm1\}^{T\setminus B}} \left| \sum_{S\subseteq U\setminus B:|S|\le\sqrt{k/\varepsilon}\cdot\log(1/\varepsilon)} \widehat{f_{B\to z}}(S)\chi_S(x)\rho^{|S|}\right|$$

for all $U : B\subseteq U\subseteq\mathcal{S}$ of size k simultaneously. We return $(U,\widetilde{c_U})$ for the set U with maximal $\widetilde{c_U}$.

Complexity *The procedure uses $\log(1/\delta)2^{\widetilde{O}(\sqrt{k/\varepsilon})}$ queries and runs in time $\log(1/\delta)2^{k\cdot\widetilde{O}(1/\varepsilon)}$.*

Correctness *In the case where all estimates are ε-accurate, the following holds. If $B\subseteq T$ satisfies Equation (2.5), the above procedure would return $(U,\widetilde{c_U})$ with $\widetilde{c_U}\ge\mathsf{corr}(f,\mathcal{J}_T)-3.2\varepsilon$. Moreover, regardless of whether T and B satisfy Equation (2.5), we have $\widetilde{c_U}\le\mathsf{corr}(f,\mathcal{J}_U)+2\varepsilon$.*

Proof. First we show that we can estimate all c_U up to error ε simultaneously with high probability using the aforementioned query complexity and running time. We sample $t=O(\log(1/\delta)/\varepsilon^2)$ different $z\in\{\pm1\}^B$, and estimate for each value of z the Fourier coefficients of $\widehat{f_{B\to z}}(S)$ of all sets $S\subseteq\mathcal{S}$ of size at most $\zeta=\sqrt{k/\varepsilon}\cdot\log(\frac{2}{\varepsilon})$ up to additive error $\varepsilon/\binom{k}{\le\zeta}=2^{-\widetilde{\Omega}(\sqrt{k/\varepsilon})}$ with probability $1-\frac{\delta}{t\cdot\binom{k}{\le\zeta}}$, which is possible via Fact 1.9.1 with $\log(1/\delta)2^{\widetilde{O}(\sqrt{k/\varepsilon})}$ queries. Fact 1.9.1 guarantees that with probability $1-\delta$ for all sampled z, all estimated low-degree Fourier coefficients are within the additive error bound, in which case we have estimates for all c_U up to error ε simultaneously with probability $1-\delta$.

Next, we show the correctness of the procedure. On the one hand, in the assumed case, i.e., that T satisfies $\mathbf{E}_z\left[\sum_{S\subseteq T\setminus B,|S|\geq \mathsf{thr}} \widehat{f_{B\to z}}(S)^2\right] \leq \frac{\varepsilon^2}{4}$, we will have by Claim 2.3.6 and Claim 2.3.7 that

$$c_T \geq \mathsf{corr}(f,\mathcal{J}_T) - 2.2\varepsilon \tag{2.10}$$

Since we output the set U with maximal $\widetilde{c_U}$, and since all estimates are correct up to ε we know that we output U with

$$\widetilde{c_U} \geq \widetilde{c_T} \geq c_T - \varepsilon. \tag{2.11}$$

Combining Equations (2.10) and (2.11) together we get

$$\widetilde{c_U} \geq c_T - \varepsilon \geq \mathsf{corr}(f,\mathcal{J}_T) - 3.2\varepsilon.$$

We move to prove the furthermore part, i.e., that $\widetilde{c_U} \leq \mathsf{corr}(f,\mathcal{J}_U) + 2\varepsilon$ regardless of whether T and B satisfy Equation (2.5). We start by showing that for any set U (whatsoever) we have that $\mathsf{corr}(f,\mathcal{J}_U) \geq c_U - \varepsilon$. Indeed, by Claim 2.3.6 we have

$$c_U \approx_\varepsilon \mathop{\mathbf{E}}_{\substack{z\in\{\pm1\}^B\\ x\in\{\pm1\}^{U\setminus B}}} \left|\sum_{S\subseteq U\setminus B} \widehat{f_{B\to z}}(S)\chi_S(x)\rho^{|S|}\right|$$

and since the noise operator can only reduce ℓ_1-norm (see Fact 1.9.4), we see that for all $z\in\{\pm1\}^B$ it holds that

$$\mathop{\mathbf{E}}_{x\in\{\pm1\}^{U\setminus B}} \left|\sum_{S\subseteq U\setminus B} \widehat{f_{B\to z}}(S)\chi_S(x)\rho^{|S|}\right| \leq \mathop{\mathbf{E}}_{x\in\{\pm1\}^{U\setminus B}} \left|\sum_{S\subseteq U\setminus B} \widehat{f_{B\to z}}(S)\chi_S(x)\right|$$

Thus,

$$\begin{aligned} c_U &\leq \varepsilon + \mathop{\mathbf{E}}_{\substack{z\in\{\pm1\}^B\\ x\in\{\pm1\}^{U\setminus B}}} \left|\sum_{S\subseteq U\setminus B} \widehat{f_{B\to z}}(S)\chi_S(x)\rho^{|S|}\right| \\ &\leq \varepsilon + \mathop{\mathbf{E}}_{\substack{z\in\{\pm1\}^B\\ x\in\{\pm1\}^{U\setminus B}}} \left|\sum_{S\subseteq U\setminus B} \widehat{f_{B\to z}}(S)\chi_S(x)\right| = \varepsilon + \mathsf{corr}(f,\mathcal{J}_U) \end{aligned}$$

Since $|c_U - \widetilde{c_U}| \leq \varepsilon$, we get that $\widetilde{c_U} \leq c_U + \varepsilon \leq \mathsf{corr}(f,\mathcal{J}_U) + 2\varepsilon$. □

After phase one, we can apply Lemma 2.3.8 to each B from phase one, and get a set $U_B : B \subseteq U_B \subseteq \mathcal{S}$ of size k, along with an estimate of the correlation of f to $\mathcal{J}_{U_B}$. This leads to the proof of Theorem 1.8.1 which we restate next.

Theorem 2.3.9. *Given a Boolean function $f:\{\pm1\}^n \to \{\pm1\}$, it is possible to estimate the distance of f from the class of k-juntas to within additive error ε with probability $2/3$ using $2^{\widetilde{O}(\sqrt{k/\varepsilon})}$ adaptive queries to f. In particular, when ε is constant, this yields a $2^{\widetilde{O}(\sqrt{k})}$-query algorithm. However, the algorithm still requires $\exp(k/\varepsilon)$ time.*

Proof. Let $\varepsilon_0 = \varepsilon/6$

1. We first apply the result of [DMN19] to reduce the down to only $\mathsf{poly}(k, 1/\varepsilon_0)$ coordinates. This incurs a loss in correlation of at most ε_0, and fails with probability at most δ_1, which we can set to be $1/20$, by Corollary 2.2.3.

2. Next, we apply our Theorem 1.8.3, which reduces the number of oracles we have to consider down to $O(k/\varepsilon_0^2)$, incurs an additive loss in correlation of at most ε_0, and fails with probability at most $\delta_2 = 1/20$.

3. Then, we run phase 1 of our algorithm, which fails with probability at most $\delta_3 = 1/20$ by Lemma 2.3.3.

4. Finally, we apply Lemma 2.3.8 to every B outputted by Algorithm 9 to get a set U_B and an estimate $\widetilde{C_{U_B}}$ for the correlation of f with $\mathcal{J}_{U_B}$ We iterate on all sets B returned by phase-1 and return U_B with the highest estimate of correlation.

   There are $\ell = O(\frac{1}{\varepsilon_0^2})^{3\sqrt{k/\varepsilon_0}+\log(2/\delta_3)} = 2^{\widetilde{O}(\sqrt{k/\varepsilon_0})}$ branches, and thus if we apply the algorithm from lemma 2.3.8 with $\delta = 1/(20\ell)$, we get that all this step fail with probability at most $1/20$ by a union bound.

By a union bound, each of these steps succeeds with probability at least $1 - 4/20 \geq 2/3$. In the case all steps succeeds, we return a set U with $\widetilde{c_U} \geq \mathsf{corr}(f, \mathcal{J}_{n,k}) - 5.2\varepsilon_0$. In addition, the moreover part in Lemma 2.3.8 guarantees that $\widetilde{c_U} \leq \mathsf{corr}(f, \mathcal{J}_U) + 2\varepsilon_0 \leq \mathsf{corr}(f, \mathcal{J}_{n,k}) + 2\varepsilon_0$. We get that the returned value is within $5.2\varepsilon_0 < \varepsilon$ of $\mathsf{corr}(f, \mathcal{J}_{n,k})$. Finally, since $\mathsf{dist}(f, \mathcal{J}_{n,k}) = \frac{1+\mathsf{corr}(f,\mathcal{J}_{n,k})}{2}$ we get that $\frac{1+\widetilde{c_U}}{2}$ is an $\varepsilon/2$-accurate approximation of $\mathsf{dist}(f, \mathcal{J}_{n,k})$. Finally, we note that the query complexities of phase 1 and phase 2 are both $2^{\widetilde{O}(\sqrt{k/\varepsilon})}$, but the runtime is exponential due to lemma 2.3.8. □

Finally, we mention that if our goal is not to estimate to correlation with the nearest k-junta to f, but rather to simply estimate the most amount of Fourier mass any subset of k variables contains, then we have the following theorem with an improved dependence on ε:

Theorem 2.3.10. *Given a Boolean function $f : \{\pm 1\}^n \to \{\pm 1\}$, it is possible to estimate the most mass any subset of at most k variables of f has to within additive error ε with probability $2/3$ using $2^{\widetilde{O}(\sqrt{k}\log(1/\varepsilon))}$ adaptive queries to f. In particular, when ε is constant, this yields a $2^{\widetilde{O}(\sqrt{k})}$-query algorithm. However, the algorithm still requires $\exp(k\log(1/\varepsilon))$ time.*

We leave the proof of this theorem, which involves simple modifications to the algorithm presented in this section, to Section C.

2.3.3 Proof of Theorem 2.3.1

We now present the proof of Theorem 2.3.1.

Proof of Theorem 2.3.1. The proof is very similar to the previous proof of Theorem 2.2.11, so we explain how to modify it to this case.

We express λ_U in terms of the Fourier spectrum of f.

$$\begin{aligned}\lambda_U &= \sum_{m=0}^{2|U|\log(10k)} \sum_{S:S\supseteq U} \widehat{f}(S)^2 \cdot \mathop{\mathbf{Pr}}_{J\subseteq_{p^m}[\ell]}[S\cap J = U] \\ &= \sum_{m=0}^{2|U|\log(10k)} \sum_{S:S\supseteq U} \widehat{f}(S)^2 \cdot \mathop{\mathbf{Pr}}_{J\subseteq_{p^m}[\ell]}[|S\cap J| = |U|]\cdot \frac{1}{\binom{|S|}{|U|}} \\ &= \sum_{S:S\supseteq U} \frac{\widehat{f}(S)^2}{\binom{|S|}{|U|}} \cdot \sum_{m=0}^{2|U|\log(10k)} \mathop{\mathbf{Pr}}_{J\subseteq_{p^m}[\ell]}[|S\cap J| = |U|]\end{aligned}$$

It suffices to show that for any non-empty set S of size at least $|U|$ and at most k it holds that

$$\sum_{m=0}^{2|U|\log(10k)} \mathop{\mathbf{Pr}}_{J\subseteq_{p^m}[\ell]}[|S\cap J| = |U|] \in [1/2, 3]\ . \tag{2.12}$$

Again, we can analyze the sum on the left hand side of Equation (2.12) as the expected final value of X in the following random process:

```
1 $X \leftarrow 0$
2 for $i = 1, 2, \dots, 2|U|\log(10k)$ do
3     if $|S| < |U|$ then
4         halt!
5     if $|S| = |U|$ then
6         increase $X$
7     Sample $J_i \subseteq_p [\ell]$
8     $S \leftarrow S \cap J_i$
```

By symmetry the expected value depends only on the size of the initial set S. As before, we denote by F_t its expected value starting with a set S of size t with an infinite horizon, and $F_t^{(i)}$ as the expected value of X at the end of the above process with finite horizon i. We start by analyzing $F_{|U|}$. In this case, X is a geometric random variable with stopping probability $1 - p^{|U|}$. Thus, its expectation is

$$F_{|U|} = 1/(1 - p^{|U|}) = 1/(1 - (1 - 1/2|U|)^{|U|}) \in [2, 3].$$

This implies that $F_{|U|}^{(2|U|\log(10k))} \le F_{|U|} \le 3$. For $t > |U|$ in the infinite horizon case we have the recurrence

$$F_t = \sum_{a=0}^{t} F_a \cdot \mathbf{Pr}[\mathrm{Bin}(t,p) = a] = \sum_{a=|U|}^{t-1} F_a \cdot \mathbf{Pr}[\mathrm{Bin}(t,p) = a] + F_t \cdot \mathbf{Pr}[\mathrm{Bin}(t,p) = t] \tag{2.13}$$

or equivalently

$$F_t \cdot \mathbf{Pr}[\mathrm{Bin}(t,p) < t] = \sum_{a=|U|}^{t-1} F_a \cdot \mathbf{Pr}[\mathrm{Bin}(t,p) = a] \tag{2.14}$$

We prove by induction that for $t \geq |U|$ it holds that $F_t \leq F_{|U|}$. The claim clearly holds for $t = |U|$. For $t > |U|$ we can apply induction and get

$$F_t \cdot \mathbf{Pr}[\mathrm{Bin}(t,p) < t] \leq \sum_{a=|U|}^{t-1} F_{|U|} \cdot \mathbf{Pr}[\mathrm{Bin}(t,p) = a] \leq F_{|U|} \cdot \mathbf{Pr}[\mathrm{Bin}(t,p) < t],$$

and thus $F_t \leq F_{|U|}$. This immediately implies that $F_t^{(2|U|\log(10k))} \leq F_t \leq 3$. On the other hand we prove that $F_t^{(2|U|\log(10k))} \geq 1/2$ as long as $t \leq k$. To do so, we once again introduce the indicator random variable $Y_t^{(i)}$, where $t = |S|$, and which equals 1 if $|S| = |U|$ at some point during the above process before iteration i. We note that $Y_t^{(2|U|\log(10k))}$ is a lower bound for the value of X in the above process, and Y_t is a lower bound for the value of X at the end of the infinite horizon process. We note that the case $|U| = 1$ was already lower bounded in Section 2.2.5, where it was shown that $\mathbf{E}[Y_t^{(\log(10k))}] \geq 1/2$, and therefore $\mathbf{E}[Y_t^{(2|U|\log(10k))}] \geq 1/2$. It remains to show that the $\mathbf{E}[Y_t^{(2|U|\log(10k))}] \geq 1/2$ is true for any set $|U| \geq 2$.

First, we show that $\mathbf{Pr}[\mathrm{Bin}(t,p) < |U|] \leq \frac{1}{2}\mathbf{Pr}[\mathrm{Bin}(t,p) = |U|]$. Towards this goal, it would suffice to prove that $3 \leq \mathbf{Pr}[\mathrm{Bin}(t,p) = i+1]/\mathbf{Pr}[\mathrm{Bin}(t,p) = i]$ for $i < |U|$ and $t \geq |U|+1$. This would suffice since in this case

$$\sum_{i=0}^{|U|-1} \mathbf{Pr}[\mathrm{Bin}(t,p) = i] \leq \sum_{i=0}^{|U|-1} \frac{3^i}{3^{|U|}} \mathbf{Pr}[\mathrm{Bin}(t,p) = |U|] \leq \frac{1}{2} \cdot \mathbf{Pr}[\mathrm{Bin}(t,p) = |U|].$$

Indeed, The ratio between the two aforementioned probabilities is

$$\frac{\mathbf{Pr}[\mathrm{Bin}(t,p) = i+1]}{\mathbf{Pr}[\mathrm{Bin}(t,p) = i]} = \frac{\binom{t}{i+1}}{\binom{t}{i}} \cdot \frac{p^{i+1}(1-p)^{t-(i+1)}}{p^i(1-p)^{t-i}} = \frac{t-i}{i+1} \cdot \frac{p}{1-p} \geq \frac{2}{|U|} \cdot \frac{1-1/2|U|}{1/2|U|} = \frac{2-1/|U|}{1/2} \geq 3$$

as needed. Now, we claim that $\mathbf{E}[Y_t] = \mathbf{Pr}[Y_t = 1] \geq 2/3$ for all $t \geq 1$. The base case of $t = 1$ is certainly true. Assuming we have $\mathbf{Pr}[\mathrm{Bin}(t,p) < |U|] \leq \frac{1}{2}\mathbf{Pr}[\mathrm{Bin}(t,p) = |U|]$ we have

$$\begin{aligned}
\mathbf{E}[Y_t] \cdot \mathbf{Pr}[Bin(t,p) < t] &= \sum_{a=|U|}^{t-1} \mathbf{E}[Y_a] \cdot Pr[\mathrm{Bin}(t,p) = a] \\
&\geq \mathbf{Pr}[\mathrm{Bin}(t,p) = |U|] + \sum_{a=|U|+1}^{t-1} \mathbf{Pr}[\mathrm{Bin}(t,p) = a]\,\mathbf{E}[Y_a] \\
&\geq \mathbf{Pr}[\mathrm{Bin}(t,p) = |U|] + \frac{2}{3}\mathbf{Pr}[\mathrm{Bin}(t,p) \in [|U|+1, t-1]] \\
&= \frac{2}{3}\mathbf{Pr}[\mathrm{Bin}(t,p) < t] - \frac{2}{3}\mathbf{Pr}[\mathrm{Bin}(t,p) < |U|] + \frac{1}{3}\mathbf{Pr}[\mathrm{Bin}(t,p) = |U|] \\
&\geq \frac{2}{3}\mathbf{Pr}[\mathrm{Bin}(t,p) < t]
\end{aligned}$$

which implies that $\mathbf{E}[Y_t] \geq 2/3$. Finally, let A be the event that $S = \emptyset$ by iteration $2|U|\log(10k)$, and note that

$$\begin{aligned}
\mathbf{Pr}[A] &= \mathbf{Pr}[\mathrm{Bin}(|S|, (1 - \tfrac{1}{2|U|})^{2|U|\log(10k)}) = 0] \\
&\geq \mathbf{Pr}[\mathrm{Bin}(k, e^{-\log(10k)}) = 0] = \mathbf{Pr}[\mathrm{Bin}(k, \tfrac{1}{10k}) = 0] \geq 0.9
\end{aligned}$$

as was shown in the proof for Theorem 2.2.11 in Section 2.2.5. Finally, we claim that for all $t \geq 2$ we have that $\mathbf{Pr}[Y_t^{(2|U|\log 10k)}] \geq 1/2$. Indeed, we have that

$$\mathbf{Pr}[Y_t^{(2|U|\log 10k))} = 1] \geq \mathbf{Pr}[Y_t = 1] - \mathbf{Pr}[\overline{A}] \geq \tfrac{2}{3} - 0.1 \geq \tfrac{1}{2}.$$

as desired, provided $|S| \leq k$. □

2.4 Conclusions and Open Problems

We conclude by mentioning some future research directions. First, we believe some of the techniques discussed in this thesis could lead to other interesting work in property testing, learning theory, or Boolean function analysis in general. In particular, the procedure in Algorithm 5 makes use of a random process to get access to an underlying junta, a subprocedure that could be useful in other learning or testing algorithms. In addition, we are able to approximate the quantities $\mathbf{NInf}_i$ and $\mathbf{NInf}_U$, that serve as key steps in our algorithms. These quantities seems natural on their own, and would likely find further applications in the analysis of Boolean functions. In particular, they seem to capture more accurately the intuition that “influences measures the importance of coordinates”. While the total influence of a Boolean function can be any number between $\mathbf{Var}[f]$ and $n \cdot \mathbf{Var}[f]$ the total normalized influence equals exactly $\mathbf{Var}[f]$, and thus normalized influences can be seen as a distribution of the variance among the coordinates.

Interestingly, our algorithms strongly resemble certain quantum algorithms and indeed were inpsired by them. In particular, the sampling of coordinates is done through the Fourier distribution, a process which can be done much more efficiently with a quantum algorithm, as discussed in Section 1.5.2. This idea was leveraged in [AS07, ABRdW16] to provide fast quantum algorithms for testing juntas in the standard property testing regime. Indeed, if the nearest k-junta to f has its mass on higher levels (say above $\sqrt{k}$ or even $k/2$), then Fourier sampling is extremely effective and provides a cleaner way of sampling subsets according to the Fourier distribution than the related classical technique we provided in Section 2.3. However, the issue arises when the nearest k-junta has Fourier mass on lower levels (below $\sqrt{k}$ or $\log k$ or even a constant, for example). In this case, it is not clear to us how quantum algorithms provide any advantage over classical ones.

We now list some more concrete open questions that we hope can be resolved in the near future.

Open Question 2.4.1. *Can we close the gap between the $\Omega(k^{1/3})$ lower bound and $\widetilde{O}(\sqrt{k/\varepsilon})$ upper bound for the adaptive quantum junta testers, as exhibited in Table 1.2? Moreover, can we close the gap in the nonadaptive case?*

Open Question 2.4.2. *Can quantum Fourier sampling techniques be applied in a more clever way to give faster algorithms in the tolerant testing regime?*

Open Question 2.4.3. *Is it possible improve the lower bound of $2^{k^{0.499}}$ for nonadaptive tolerant junta testing given in [PRW19]? In particular, can we improve it using the erasure-resilient model, or do we need different techniques, or does there actually exist a matching upper bound? A more modest goal would be exhibiting any nontrivial nonadaptive tolerant junta tester.*

Open Question 2.4.4. *Can we provide any improved lower bounds for the* adaptive *tolerant testing of juntas? Existing techniques [LW19, PRW19] are only for nonadaptive algorithms, and any lower bound of $\omega(k \log k)$ would be interesting.*

Open Question 2.4.5. *Is the purely exponential time complexity inherent for the tolerant testing of k-juntas? We suspect the answer is yes, at least under reasonable complexity-theoretic assumptions.*

Finally, a clear open question is how good of a lower bound one can prove on the query complexity of the tolerant junta testing problem. Our main result Theorem 1.8.1, rules out strictly exponential-in-k query lower bounds for k-junta distance approximation. [PRW20] proved a non-adaptive query complexity lower bound of 2^{k^η} for $(k, k, \varepsilon_1, \varepsilon_2)$-tolerant junta testing (given a particular choice of $0 < \varepsilon_1 < \varepsilon_2 < 1/2$), for any $0 < \eta < 1/2$. While this is quite close to our upper bound of $2^{\widetilde{O}(\sqrt{k})}$, our algorithm is highly adaptive, while the lower bound due to [PRW20] applies only to nonadaptive algorithms.

Acknowledgements

We thank Anindya De, Shafi Goldwasser, Amit Levi, and Orr Paradise for very helpful discussions.

Bibliography

[ABRdW16] Andris Ambainis, Aleksandrs Belovs, Oded Regev, and Ronald de Wolf. Efficient quantum algorithms for (gapped) group testing and junta testing. In *SODA*, pages 903–922. SIAM, 2016. 15, 16, 19, 20, 58

[ACCL07] Nir Ailon, Bernard Chazelle, Seshadhri Comandur, and Ding Liu. Estimating the distance to a monotone function. *Random Structures & Algorithms*, 31(3):371–383, 2007. 9

[ALM$^+$92] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and hardness of approximation problems. In *FOCS*, pages 14–23. IEEE Computer Society, 1992. 8

[AS07] Alp Atici and Rocco A. Servedio. Quantum algorithms for learning and testing juntas. *Quantum Inf. Process.*, 6(5):323–348, 2007. 15, 16, 19, 20, 58

[BBC$^+$01] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. *J. ACM*, 48(4):778–797, 2001. 15

[BBM12] Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. *Comput. Complex.*, 21(2):311–358, 2012. 17, 18, 20

[BCE$^+$19] Eric Blais, Clément L. Canonne, Talya Eden, Amit Levi, and Dana Ron. Tolerant junta testing and the connection to submodular optimization and function isomorphism. *ACM Trans. Comput. Theory*, 11(4):24:1–24:33, 2019. 5, 11, 21, 22, 25, 26

[BCH$^+$96] Mihir Bellare, Don Coppersmith, Johan Håstad, Marcos A. Kiwi, and Madhu Sudan. Linearity testing in characteristic two. *IEEE Trans. Inf. Theory*, 42(6):1781–1795, 1996. 12

[BFL91] László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Comput. Complex.*, 1:3–40, 1991. 8

[BFNR08] Harry Buhrman, Lance Fortnow, Ilan Newman, and Hein Röhrig. Quantum property testing. *SIAM J. Comput.*, 37(5):1387–1400, 2008. 15

[BGMdW13] Harry Buhrman, David García-Soriano, Arie Matsliah, and Ronald de Wolf. The non-adaptive query complexity of testing k-parities. *Chic. J. Theor. Comput. Sci.*, 2013, 2013. 18

[BGS98] Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, pcps, and nonapproximability-towards tight results. *SIAM J. Comput.*, 27(3):804–915, 1998. 12

[BHL95] Avrim Blum, Lisa Hellerstein, and Nick Littlestone. Learning in the presence of finitely or infinitely many irrelevant attributes. *J. Comput. Syst. Sci.*, 50(1):32–40, 1995. 13

[BJ99] Nader H. Bshouty and Jeffrey C. Jackson. Learning DNF over the uniform distribution using a quantum example oracle. *SIAM J. Comput.*, 28(3):1136–1153, 1999. 15, 19

[Bla08] Eric Blais. Improved bounds for testing juntas. In *APPROX-RANDOM*, volume 5171 of *Lecture Notes in Computer Science*, pages 317–330. Springer, 2008. 11, 14, 16, 18, 19, 20, 22, 23

[Bla09] Eric Blais. Testing juntas nearly optimally. In *STOC*, pages 151–158. ACM, 2009. 11, 13, 14, 16, 18, 19, 20, 21, 22

[Bla10] Eric Blais. Testing juntas: A brief survey. In *Property Testing*, volume 6390 of *Lecture Notes in Computer Science*, pages 32–40. Springer, 2010. 9

[Bla12] Eric Blais. Testing properties of boolean functions. *PhD Thesis*, 2012. 14

[BLR90] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. In *STOC*, pages 73–83. ACM, 1990. 7, 12

[BLT20] Guy Blanc, Jane Lange, and Li-Yang Tan. Testing and reconstruction via decision trees. *CoRR*, abs/2012.08735, 2020. 9

[Bsh19a] Nader H. Bshouty. Almost optimal distribution-free junta testing. In *Computational Complexity Conference*, volume 137 of *LIPIcs*, pages 2:1–2:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. 16

[Bsh19b] Nader H. Bshouty. Almost optimal testers for concise representations. *CoRR*, abs/1904.09958, 2019. 15, 16

[Can15] Clément L. Canonne. A survey on distribution testing: Your data is big. but is it blue? *Electron. Colloquium Comput. Complex.*, 22:63, 2015. 9

[CFGM12] Sourav Chakraborty, Eldar Fischer, David García-Soriano, and Arie Matsliah. Junto-symmetric functions, hypergraph isomorphism and crunching. In *Computational Complexity Conference*, pages 148–158. IEEE Computer Society, 2012. 21

[CG04] Hana Chockler and Dan Gutfreund. A lower bound for testing juntas. *Inf. Process. Lett.*, 90(6):301–305, 2004. 17, 18, 19, 20

[CST$^+$18] Xi Chen, Rocco A. Servedio, Li-Yang Tan, Erik Waingarten, and Jinyu Xie. Settling the query complexity of non-adaptive junta testing. *J. ACM*, 65(6):40:1–40:18, 2018. 19, 20, 23

[DLM$^+$07] Ilias Diakonikolas, Homin K. Lee, Kevin Matulef, Krzysztof Onak, Ronitt Rubinfeld, Rocco A. Servedio, and Andrew Wan. Testing for concise representations. In *FOCS*, pages 549–558. IEEE Computer Society, 2007. 15, 21

[DLM$^+$08] Ilias Diakonikolas, Homin K. Lee, Kevin Matulef, Rocco A. Servedio, and Andrew Wan. Efficiently testing sparse GF(2) polynomials. In *ICALP (1)*, volume 5125 of *Lecture Notes in Computer Science*, pages 502–514. Springer, 2008. 15

[DMN19] Anindya De, Elchanan Mossel, and Joe Neeman. Junta correlation is testable. In *FOCS*, pages 1549–1563. IEEE Computer Society, 2019. 11, 15, 21, 22, 25, 26, 30, 33, 34, 38, 43, 55, 67, 70

[FKR$^+$04] Eldar Fischer, Guy Kindler, Dana Ron, Shmuel Safra, and Alex Samorodnitsky. Testing juntas. *J. Comput. Syst. Sci.*, 68(4):753–787, 2004. 12, 13, 16, 17, 18, 21, 22

[GE03] Isabelle Guyon and Andre Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, 2003. 9

[GGR96] Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 339–348. IEEE Computer Society, 1996. 8, 9

[Gol11] Oded Goldreich. Introduction to testing graph properties. In *Studies in Complexity and Cryptography*, volume 6650 of *Lecture Notes in Computer Science*, pages 470–506. Springer, 2011. 9

[Gol17] Oded Goldreich. *Introduction to Property Testing.* Cambridge University Press, 2017. 8

[HK07] Shirley Halevy and Eyal Kushilevitz. Distribution-free property-testing. *SIAM J. Comput.*, 37(4):1107–1138, 2007. 15

[H0̊1] Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, July 2001. 67

[ITW21] Vishnu Iyer, Avishay Tal, and Michael Whitmeyer. Junta distance approximation with sub-exponential queries. *Electron. Colloquium Comput. Complex.*, 28:4, 2021. 1, 15, 19, 25, 26

[KN97] Eyal Kushilevitz and Noam Nisan. *Communication complexity.* Cambridge University Press, 1997. 17

[KR00] Michael J. Kearns and Dana Ron. Testing problems with sublearning sample complexity. *J. Comput. Syst. Sci.*, 61(3):428–456, 2000. 8

[LCS$^+$19] Zhengyang Liu, Xi Chen, Rocco A. Servedio, Ying Sheng, and Jinyu Xie. Distribution-free junta testing. *ACM Trans. Algorithms*, 15(1):1:1–1:23, 2019. 15, 16

[LW19] Amit Levi and Erik Waingarten. Lower bounds for tolerant junta and unateness testing via rejection sampling of graphs. In *ITCS*, volume 124 of *LIPIcs*, pages 52:1–52:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. 22, 58

[MOS03] Elchanan Mossel, Ryan O'Donnell, and Rocco A. Servedio. Learning juntas. In *STOC*, pages 206–212. ACM, 2003. 9

[O'D14] Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014. 7, 8, 10, 12, 69

[PRR04] Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *Electron. Colloquium Comput. Complex.*, (010), 2004. 8, 17, 21

[PRS01] Michal Parnas, Dana Ron, and Alex Samorodnitsky. Proclaiming dictators and juntas or testing boolean formulae. In *RANDOM-APPROX*, volume 2129 of *Lecture Notes in Computer Science*, pages 273–284. Springer, 2001. 12

[PRW19] Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Erik Waingarten. Approximating the distance to monotonicity of boolean functions. *CoRR*, abs/1911.06924, 2019. 4, 9, 23, 24, 25, 26, 58

[PRW20] Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Erik Waingarten. Approximating the distance to monotonicity of boolean functions. In *SODA*, pages 1995–2009. SIAM, 2020. 9, 59

[RY20] Anup Rao and Amir Yehudayoff. *Communication Complexity: and Applications*. Cambridge University Press, 2020. 17

[Sag18] Mert Saglam. Near log-convexity of measured heat in (discrete) time and consequences. In *FOCS*, pages 967–978. IEEE Computer Society, 2018. 15, 18, 20, 26

[Ser10] Rocco A. Servedio. Testing by implicit learning: A brief survey. In *Property Testing*, volume 6390 of *Lecture Notes in Computer Science*, pages 197–210. Springer, 2010. 15

[Tre04] Luca Trevisan. Inapproximability of combinatorial optimization problems. *Electron. Colloquium Comput. Complex.*, (065), 2004. 8

[Val12] Gregory Valiant. Finding correlations in subquadratic time, with applications to learning parities and juntas. In *FOCS*, pages 11–20. IEEE Computer Society, 2012. 9

[Yao77] Andrew Chi-Chih Yao. Probabilistic computations: Toward a unified measure of complexity (extended abstract). In *FOCS*, pages 222–227. IEEE Computer Society, 1977. 65

[Zha19] Xiaojin Zhang. Near-optimal algorithm for distribution-free junta testing. *CoRR*, abs/1911.10833, 2019. 15, 16

Appendix

A Minimax

This section is devoted to rederiving the minimax theorem via the separating hyperplane theorem, and explaining its connection to lower bounds for randomized algorithms via Yao's principle [Yao77].[2] We shall go about this backwards, first using the minimax theorem to derive Yao's principle, and then we will circle back and prove the minimax theorem.

A.1 Yao's Principle from Minimax

We first state the minimax theorem in the setting of two player games. Consider player one having a choice over m "pure" strategies, and player two have a choice over n pure strategies. If player 1 plays strategy i and player 2 plays strategy j, then the "payoff" will be P_{ij}. The goal of player one is to maximize the expected payoff, while the goal of player two is to minimize it. We can encode all the possible payoffs into a matrix P, and then associate each "pure" strategy with a corresponding standard basis vector. For example, if player one uses pure strategy i, while player two uses pure strategy j, then the the payoff is $e_i^T P e_j = P_{ij}$. However, we also allow the players to use "mixed" strategies, i.e. we can replace any basis vector e_i with a vector p such that $\sum_i p_i = 1$ and all the entries are nonnegative. We are now ready to state the minimax theorem, which says that

$$\min_p \max_q q^T P p = \max_q \min_p q^T P p. \tag{2.15}$$

Let us pause and interpret what this is saying. Somewhat counterintuitively, the LHS of the above equation corresponds to player 1 going second (despite the maximization being on the "inside"). This is because the maximization occurs for a fixed strategy p of player two. Player two (on the left hand side), must try to come up with a strategy that minimizes the payoff, assuming (correctly) that player 1 will act optimally after they have made their choice. Likewise, the RHS corresponds to player two going first. With this perspective/interpretation of the above, it is clear that the left hand side is greater than or equal to the right hand side. The surprising thing is the equality itself, which we prove in Section A.2.

Now, we can discuss Yao's principle. In a nutshell, Yao's principle gives an equivalence between two types of randomness within algorithms: randomness in the algorithm itself and randomness over the inputs. Fix some model of computation for computing a Boolean function f (in this thesis, the model is almost always the query model, but one can also

[2]Much of this section was taken from Ronald De Wolf's nice writeup https://homepages.cwi.nl/ rdewolf/simonlowerbound.pdf.

consider other models/complexity measures, such as communication complexity). We denote $R_\varepsilon(f)$ as the minimal complexity among all randomized algorithms that compute $f(x)$ with success probability at least $1-\varepsilon$.[3] We also let $D^\mu_\varepsilon(f)$ be the minimal complexity among all *deterministic* algorithms that compute f on at least $1-\varepsilon$ fraction of all inputs, which are weighed according to the input distribution μ. Yao's lemma says the following:

$$R_\varepsilon(f) = \max_\mu D^\mu_\varepsilon(f).$$

In words: the complexity of any randomized algorithm solving a problem is equal to the complexity of a deterministic algorithm solving the problem on the hardest possible distribution of inputs. It has proven extremely useful for proving lower bounds for randomized algorithms, since it is often much easier to analyze a deterministic algorithm (over a distribution of inputs) than a randomized one. Yao's principle is actually a direct corollary of the minimax theorem, as we will now see. First, we note that WLOG we may take the inner optimization to be over pure strategies,[4] so Equation (2.15) becomes:

$$\min_p \max_{e_i} e_i^T P p = \max_q \min_{e_j} q^T P e_j.$$

Now, we interpret this equation in the context of computation. We can think of player 1 as picking an algorithm, while player 2 picks a distribution over inputs. The pure strategies for player 1 are all deterministic algorithms with complexity at most c, and the mixed strategies of course correspond to randomized algorithms. Pure strategies for player 2 are just inputs, and randomized strategies are then distributions over inputs. Finally, the payoff matrix has $P_{ij} = 1$ if algorithm i correctly computes f on input j. Now, on the left hand side, $e_i^T P p$ is the fraction of inputs on which deterministic algorithm i is correct, where the inputs are weighed according to p, so $\max_i e_i^T P p$ is this fraction for the best possible algorithm. So the LHS gives the optimal fraction of correct inputs achievable by complexity c deterministic algorithms on input distribution p.

On the other hand, $q^T P e_j$ is the success probability on input e_j achieved by the randomized algorithm given by probability distribution q over deterministic algorithms, and $\min_{e_j} q^T P e_j$ is its success probability on the hardest input. Thus, the RHS gives the highest worst-case success probability achievable by randomized algorithms with complexity c. Since these two quantities are equal for all complexities c, and minimizing the payoff for a given complexity is equivalent to maximizing the complexity for a given payoff, we have

$$R_\varepsilon(f) = \max_p D^p_\varepsilon(f).$$

A.2 Proof of Minimax

We sketch a short proof of the minimax theorem via the separating hyperplane lemma. The separating hyperplane lemma says that if U, V are two disjoint, convex subsets of $\mathbb{R}^n$, then there exists a hyperplane with normal vector z such that for all $u \in U$ we have $\langle u, v\rangle \leq \alpha$, and for all $v \in V$ we have $\langle v, z\rangle \geq \alpha$ for some α.

[3] In practice we normally just take $\varepsilon = 1/3$, since we can always repeat the algorithm and achieve better ε by taking the majority vote.

[4] One of the rows/columns must produce the highest inner product, so it makes sense to just put all the weight/probability mass on that one column/row.

Recall it was clear in Equation (2.15) that the LHS was at least as large as the RHS, since in the LHS the "maximizer" is going first. We would like to show the other direction; namely, we want to show that $\min_p \max_q q^T Pp \leq \max_q \min_p q^T Pp$.

In order to achieve this, consider

$$D = \{Pp : \sum p_i = 1\}.$$

Recalling that WLOG we can take the inner optimizations to be over deterministic strategies, suppose towards a contradiction that there exists some c such that

$$\min_p \max_q q^T Pp < c < \max_q \min_p q^T Pp. \tag{2.16}$$

Then consider

$$C = \{x \in \mathbb{R}^m : \forall i, x_i \leq c\}.$$

The only way Equation (2.16) can happen is if $C \cap D = \emptyset$. Why? If they intersect then there is a choice for p by player two that results in $Pp \in C$, and no matter what q is, since it is a probability vector, we will have $\max_q \min_p q^T Pp \leq c$, contradicting Equation (2.16). But now, since we know C and D are two convex, disjoint sets, we can apply the aforementioned separating hyperplane lemma to conclude that there is some probability vector q such that $\langle q, d\rangle \geq c$ for every $d \in D$, since clearly $\langle q, c\rangle \leq c$ for any probability vector q. But this also contradicts (the left hand side) of Equation (2.16).

B Constructing Influential Coordinate Oracles

In this section, we summarize the technique introduced in [DMN19] for gaining access to coordinates with large low-degree influence. To do so, we will need the following operator, which was inspired by Hastad's dictator vs no-notable tester in [H01].

Definition B.1. *Let $\eta \in [-1,1]^n$ and Z_η denote the product distribution on $\{-1,1\}^n$ where the expectation of the i^{th} bit is η_i. For any $f : \{-1,1\}^n \to \mathbb{R}$, we define the operator*

$$\mathrm{Has}_\eta f(x) = \mathop{\mathbf{E}}_{y_1,y_2 \in \{\pm 1\}^n, y_3 \in Z_\eta} [f(y_1)f(y_2)f(x \cdot y_1 \cdot y_2 \cdot y_3)].$$

Its Fourier expansion can be written as:

$$\mathrm{Has}_\eta f(x) = \sum_S \widehat{f}^3(S)\chi_S(x)\eta^S.$$

where $\cdot$ denotes coordinate-wise multiplication (an XOR operation) and $\eta^S = \prod_{j\in S} \eta_j$.

A consequence of this definition is that $\mathrm{Has}_\eta f$ could have a good approximation to certain dictator function for the right choice of η. In particular, many of the η^S terms could be very small or even zero for larger S, depending on how we choose η. The following lemma is the key to making this intuition come to fruition:

Lemma B.2. *Suppose that $|\widehat{f}(\{1\})| \geq \kappa$, where $\kappa \in (0,1)$, and let $\alpha = \frac{\kappa^3}{16}$. Choose $\eta \in \{0,\alpha\}^n$ randomly so that $Pr[\eta_i = \alpha] = \kappa^2/16$, independently for every i. Then with probability at least $\Omega(\kappa^6)$, for every $x \in \{\pm 1\}^n$,*

$$|\mathrm{Has}_\eta f(x) - \widehat{f}^3(\emptyset) - \alpha \widehat{f}^3(\{1\})x_1| \leq \frac{\alpha}{4}|\widehat{f}(\{1\})|^3. \tag{2.17}$$

Proof. Let $\rho = \kappa^6/16 = Pr[\eta_i = \alpha]$. Let $\Gamma = \{i : |\widehat{f}(\{i\})| \geq \kappa^3/8\}$; since $\sum_S \widehat{f}(S)^2 < 1$, we have $|\Gamma| \leq 64\kappa^{-6}$. Let E be the event that $\eta_1 = \alpha$ and $\eta_j = 0$ for all $j \in \Gamma \setminus \{1\}$. That is, E roughly represents the event of us "zeroing out" the other influential coordinates of f. Then we have $\mathbf{Pr}[E] = p(1-p)^{|\Gamma|-1} \geq p(1-p)^{64\kappa^{-6}} = \Omega(\kappa^6)$. Therefore, it suffices to show that the inequality in the statement holds when E happens, as E happens with the desired probability. Using the fourier expansion of $\mathrm{Has}_\eta f$, we have the following:

$$\begin{aligned}\mathrm{Has}_\eta f(x) - \widehat{f}^3(\emptyset) - \alpha \widehat{f}^3(\{1\})x_1 &= \sum_{1<j\leq n} \eta_j \widehat{f}^3(j)x_j + \sum_{|S|>1} \eta^S \widehat{f}^3(S)\chi_S(x) \\ &= \sum_{j\notin\Gamma} \eta_j \widehat{f}^3(j)x_j + \sum_{|S|>1} \eta^S \widehat{f}^3(S)\chi_S(x).\end{aligned} \tag{2.18}$$

Where the second equality holds on event E. Now, we have

$$\sum_{j\notin\Gamma} \eta_j \widehat{f}^3(j)x_j \leq \alpha\frac{\kappa^3}{8}\sum_{j\notin\Gamma}\widehat{f}^2(j) \leq \frac{\alpha\kappa^3}{8},$$

since $\widehat{f}(j) < k^3/8$ for all $j \notin \Gamma$. We also have

$$\sum_{|S|>1} \eta^S \widehat{f}^3(S)\chi_S(x) \leq \alpha^2 \sum_{|S|>1} \widehat{f}^2(S)\chi_S(x) \leq \alpha^2.$$

Then by the triangle inequality we have

$$|\mathrm{Has}_\eta f(x) - \widehat{f}^3(\emptyset) - \alpha\widehat{f}^3(\{1\})x_1| \leq \frac{\alpha\kappa^3}{8} + \alpha^2 \leq \frac{\alpha\kappa^3}{4} \leq \frac{\alpha}{4}|\widehat{f}(\{1\})|^3. \quad \square$$

We have just shown that if $|\widehat{f}(i)|$ is large enough there is a good probability that:

$$|\mathrm{Has}_\eta f(x) - \widehat{f}^3(\emptyset) - \alpha\widehat{f}^3(\{i\})x_i| \leq \frac{\alpha}{4}|\widehat{f}(\{i\})|^3,$$

which means in particular:

$$\mathrm{Has}_\eta f(x) - \widehat{f}^3(\emptyset) = \alpha f^3(\{i\})x_i \cdot (1 \pm 1/4).$$

The important thing to note about the above equation is that the sign of the right hand sign is entirely determined by the sign of x_i. This means that if we take the sign of the above expression, it will be a dictator for the influential coordinate i! This is exactly what we wanted, so this is exactly what we do next:

$$g(f) = \mathsf{sgn}(\mathrm{Has}_\eta f(x) - \widehat{f}^3(\emptyset)) = \mathsf{sgn}(\mathrm{Has}_\eta f(x) - \mathbf{E}[f]^3). \tag{2.19}$$

The above function g will be our candidate dictator/oracle for our influential coordinate i. We first note that g requires randomness (queries to f) in order to estimate $\mathrm{Has}_\eta f(x)$ and $\mathbf{E}[f]$. By a simple Chernoff bound, with $O(\kappa^{-12}\log\frac{1}{\delta})$ queries to f, we can estimate both $\mathbf{E}[f]$ and $\mathrm{Has}_\eta(f)$ to additive accuracy $O(\kappa^6)$ with probability $1-\delta$. We can once again choose $\delta = 2^{-\mathsf{poly}(k)}$ (similar to the definition of the ν-oracle) to make the evaluation of g correct with very high probability while keeping our query complexity low enough.
We also note that Lemma B.2 gives dictators only with probability $\Omega(\kappa^6)$. This means

we will need a way to test whether or not our candidate oracle g is actually a dictator. To do so, we use a modification of the standard dictator/anti-dictator test presented in Chapter 7 of [O'D14], which makes 3 queries to f and accepts f with probability 1 if f is a dictator/anti-dictator, and rejects f with probability at least $O(\varepsilon)$ if f is ε-far from a dictator/anti-dictator. Then we can simply repeat many times and apply a chernoff bound to obtain the following theorem:

Theorem B.3 (Chapter 7 of [O'D14]). *There is an algorithm* ***Dictator-test*** *which given an error parameter $v > 0$ and confidence parameter $\widetilde{\delta} > 0$, makes $O(v^{-1} \log \widetilde{\delta}^{-1})$ queries to $f : \{\pm 1\}^n \to \{-1, 1\}$ and has the following properties:*

- *if $f : \{\pm 1\}^n \to \{\pm 1\}$ is a dictator or an anti-dictator, then it accepts with probability 1*
- *Any $f : \{\pm 1\}^n \to \{\pm 1\}$ which is v-far from every dictator and anti-dictator is accepted with probability at most $\widetilde{\delta}$.*

With this tool, we are now ready to formally give the algorithm for constructing our set of oracles to coordinates with large low-degree influence. See Algorithm 1 in the next page.

Lemma B.4 (Establishes Corollary 2.2.3). *Algorithm 1* ***(Construct-coordinate-oracle)*** *has the following guarantee:*

1. *The number of oracles in D is at most $\mathsf{poly}(k, \tau^{-1}, \log(1/\delta))$*
2. *The query complexity of the procedure is $v^{-1} \cdot \mathsf{poly}(k, \tau^{-1}, \log(1/\delta))$*

Proof. Since we generate $g(x)$ MT times, by the definition of M, T, the number of oracles in D is at most $\mathsf{poly}(k, \tau^{-1}, \log(1/\delta))$. On the other hand, query complexity of the algorithm is MT times the query complexity of **Dictator Test** which is $v^{-1} \cdot \mathsf{poly}(k, \tau^{-1}, \log(1/\delta))$. □

Algorithm 10: Construct-coordinate-oracle

```
Input: $f$ (target function), $k$ (arity of junta), $\delta$ (confidence parameter), $v \leq 1/8$
       (first accuracy parameter), $\tau$ second accuracy parameter
Output: An oracle $D$
/* Construct the initial oracles */
1  Let $T = Ck^2\tau^{-2}\log(1/\delta)$ and let $M = Ck^4\tau^{-7}\log(1/\delta)$;
2  Let $\widetilde{\delta} = \delta/(MT)$;
3  Initialize $D = \emptyset$;
4  repeat $T$ times
5      Sample $\rho$ according to $R_{1/k}$ (as in Definition 1.9.6)
6      repeat $M$ times
7          Sample $\eta$ (as in Lemma B.2);
8          Let $g(x) = \mathsf{sgn}(\mathrm{Has}_\eta f_{|\rho}(x) - \mathbf{E}[f_{|\rho}(x)]^3)$;
9          Apply Dictator-Test to $g$ with confidence $\widetilde{\delta}$ and accuracy $v$;
10         if Dictator Test accepts then
11             Add $g$ to $D$
/* Clear out duplicates */
12 Let $N = C\log(MT/\delta)$, and sample $x^{(1)}, ..., x^{(N)} \in \{\pm 1\}^n$ independently and
   uniformly;
13 while there exist $g \neq h \in D$ such that $|N^{-1}\sum_i g(x^{(i)})h(x^{(i)})| \geq \frac{1}{2}$ do
14     Remove $g$ from $D$
15 return $D$
```

Claim B.5 (Correctness of **Construct-Coordinate-Oracle**). *With probability $1-\delta$, there is some set $S \supseteq \{i : \mathbf{Inf}_i^{\leq k} \geq \tau^2/k\}$ such that the output of **Construct-coordinate-oracle** is an ν-oracle to S.*

The proof of the claim will be done in two parts. In the first part, we will show that for every influential coordinate i, at least one iteration of the outer loop will sample ρ for which $\widehat{f_{|\rho}}(i)$ is large. Then, for this execution of the outer loop, we will argue that at least one iteration of the inner loop will find an oracle for an coordinate i.

Fact B.6. *If X is a random variable bounded in $[0,1]$, with $\mathbf{E}[X] \geq \varepsilon$, then $\mathbf{Pr}[X \geq \varepsilon/2] \geq \varepsilon/2$.*

Fact B.7. *Let $f : \{\pm 1\}^n \to \mathbb{R}$ and let $J \subseteq [n]$. Then $\mathbf{E}_{z \in \{\pm 1\}^{\bar{J}}}[\widehat{f_{\bar{J} \to z}}(S)^2] = \sum_T \widehat{f}(T)^2 \cdot \mathbf{1}_{T \cap J = S}$.*

Claim B.8 (Claim 3.7 in [DMN19][5]). *If $\mathbf{Inf}_i^{\leq k}[f] \geq \delta$ and ρ is a $1/k$-random-restriction, then with probability $\frac{\delta}{6k}$, $|\widehat{f_{|_\rho}}(\{i\})| \geq \sqrt{\delta/6}$.*

Proof. Let $\rho = (J, z)$ be a $1/k$-random restriction, where J is the set of alive coordinates and $z \in \{\pm 1\}^{\bar{J}}$ is an assignment to the other variables. Suppose we condition on the event, denoted $\mathcal{E}$, that our random restriction keeps coordinate i alive (i.e., $i \in J$). Note that

[5] This claim appeared in a weaker form in [DMN19], and we present a version with improved parameters and a more self-contained proof here.

otherwise $\widehat{f_{\bar{J}\to z}}(\{i\}) = 0$. Conditioned on $\mathcal{E}$, we have

$$\begin{aligned}\mathop{\mathbf{E}}_{J,z}\left[\widehat{f_{\bar{J}\to z}}(\{i\})^2 \;\middle|\; \mathcal{E}\right] &= \mathop{\mathbf{E}}_{J}\left[\sum_{S} \widehat{f}(S)^2 \cdot \mathbf{1}_{S\cap J=\{i\}} \;\middle|\; \mathcal{E}\right] && \text{(Fact B.7)}\\ &= \sum_{S\ni i} \widehat{f}(S)^2 \cdot \mathbf{Pr}[S\cap J=\{i\} \mid \mathcal{E}]\\ &= \sum_{S\ni i} \widehat{f}(S)^2 \cdot (1-1/k)^{|S|-1}\\ &\geq (1-1/k)^{k-1} \cdot \sum_{\substack{S\ni i\\ |S|\leq k}} \widehat{f}(S)^2\\ &\geq \frac{1}{e}\cdot \mathbf{Inf}_i^{\leq k}[f] \geq \frac{\delta}{3}.\end{aligned}$$

Then, by Fact B.6, we have that $\mathbf{Pr}[\widehat{f_{\bar{J}\to z}}(\{i\})^2 \geq \frac{\delta}{6} \mid \mathcal{E}] \geq \frac{\delta}{6}$. Since $i \in J$ with probability $1/k$, this tells us that with probability at least $\frac{\delta}{6k}$ we have that $|\widehat{f_{\bar{J}\to z}}(i)| \geq \sqrt{\delta/6}$. □

In particular, for the above claim we care about the setting when $\delta = \frac{\tau^2}{k}$.

Assuming this lemma, a simple union bound with our choice of T in the algorithm implies that for every i with $\mathbf{Inf}_i^{\leq k}(f) \geq \tau^2/k$ there is at least one iteration of the outer loop for which $|\widehat{f|_\rho}(i)| \geq \frac{\tau}{\sqrt{6k}}$. Fixing this outer iteration, we can apply the same union bound argument to the inner loop with our choice of M since each iteration has $\Omega(\tau^6 k^{-3})$ probability of producing g that is close to some dictator. Thus it suffices to prove the above claim, and we have proved the correctness of **Construct-Coordinate-Oracle**.

C Maximum k-Subset Fourier Mass Approximation

In this section, we sketch a proof of Theorem 2.3.10, which involves simple modifications and observations about our algorithm. The main difference is that we sample from the normalized influence subdistribution at a different Fourier level – namely, we let $\mathsf{thr} := \sqrt{k}$ and $\alpha = k/\mathsf{thr} = \sqrt{k}$ in Algorithm 8 and Algorithm 9, respectively (recall that before, $\mathsf{thr} = \sqrt{\varepsilon k}$). This improves the query complexity dependence on ε in Phase 1.

Claim C.1. *The query complexity of phase one of the algorithm for constant δ (failure probability) is $2^{\widetilde{O}(\sqrt{k}\log(1/\varepsilon))}$.*

Proof. The proof is analogous to the proof of Claim 2.3.5, so we just point out the differences. We still require our Fourier coefficients to be accurate to within $1/\mathsf{poly}(k, 1/\varepsilon)$, and we require confidence $1 - O(1/\ell) = 1 - 2^{\widetilde{\Omega}(\sqrt{k}\log(1/\varepsilon))}$. However, now our branching process now has depth only $O(\sqrt{k})$, so we need only repeat this $O(\ell) = 2^{\widetilde{O}(\sqrt{k}\log(1/\varepsilon))}$ times, which yields the improved query complexity. □

In Phase 2, we argue that it is not necessary to apply a noise operator in order to only consider Fourier mass below level thr after Phase 1. Recall that we applied this noise operator in Section 2.3.2 in order to deal with the case that a particular U satisfied

$$\mathop{\mathbf{E}}_{z\in\{\pm1\}^B}\Bigg[\sum_{\substack{S\subseteq U\setminus B,\\ |S|>\mathsf{thr}}} \widehat{f_{B\to z}}(S)^2\Bigg] > \varepsilon^2/4. \tag{2.20}$$

If this happened, then we could not rule out the possibility that taking the low-degree part of f within U gives an *overestimate* to the correlation with the best k-junta. However, now we are not concerned with the junta correlation, but rather which set has the most mass, so we claim we do not have to worry about this possibility anymore. To see this, suppose we have identified $B\subseteq U$, and note that

$$\begin{aligned}\sum_{S\subseteq U}\widehat{f}(S)^2 &= \mathop{\mathbf{E}}_{x}[f(x)f_{\mathsf{avg},U}(x)]\\ &= \mathop{\mathbf{E}}_{z\in\{\pm1\}^B}\Big[\mathop{\mathbf{E}}_{x}[f_{B\to z}(x)(f_{\mathsf{avg},U})_{B\to z}(x)]\Big]\\ &= \mathop{\mathbf{E}}_{z}\Bigg[\sum_{\substack{S\subseteq U\\ |S|\le\mathsf{thr}}}\widehat{f_{B\to z}}(S)^2+\sum_{\substack{S\subseteq U\\ |S|>\mathsf{thr}}}\widehat{f_{B\to z}}(S)^2\Bigg]\\ &\ge \mathop{\mathbf{E}}_{z}\Bigg[\sum_{\substack{S\subseteq U\\ |S|\le\mathsf{thr}}}\widehat{f_{B\to z}}(S)^2\Bigg].\end{aligned}$$

Therefore, we no longer have to apply any noise operator, which negates the necessity of Claim 2.3.6 and Claim 2.3.7. It therefore suffices in Lemma 2.3.8 to estimate the mass of each set, rather than the correlation, as

$$m_U = \mathop{\mathbf{E}}_{z}\Bigg[\sum_{\substack{S\subseteq U\\ |S|\le\mathsf{thr}}}\widehat{f_{B\to z}}(S)^2\Bigg].$$

To do so, as in the proof of Lemma 2.3.8 we let $t = O(\log(1/\delta)/\varepsilon^2)$ be the number of random samples of z we take. Then we estimate all the Fourier coefficients below level thr. This requires estimating $\widehat{f}(S)$ for all $S\subseteq\mathcal{S}$ of size at most thr up to additive error $\varepsilon/\binom{k}{\le\kappa} = 2^{\widetilde{\Omega}(\sqrt{k}\log(1/\varepsilon))}$ with probability $1-\frac{\delta}{t\cdot\binom{k}{\le\kappa}}$, which is possible via Fact 1.9.1 with $\log(1/\delta)2^{\widetilde{O}(\sqrt{k}\log(1/\varepsilon))}$ queries. The rest of our argument and algorithm is exactly the same as in Section 2.3.