Adaptivity Is Exponentially Powerful for Testing Monotonicity of Halfspaces*†

Xi Chen[1], Rocco A. Servedio[2], Li-Yang Tan[3], and Erik Waingarten[4]

1 **Columbia University, New York, NY, USA**
xichen@cs.columbia.edu
2 **Columbia University, New York, NY, USA**
rocco@cs.columbia.edu
3 **Toyota Technological Institute, Chicago, IL, USA**
liyang@cs.columbia.edu
4 **Columbia University, New York, NY, USA**
eaw@cs.columbia.edu

Abstract

We give a $\mathrm{poly}(\log n, 1/\epsilon)$-query adaptive algorithm for testing whether an unknown Boolean function $f\colon \{-1,1\}^n \to \{-1,1\}$, which is promised to be a halfspace, is monotone versus ϵ-far from monotone. Since non-adaptive algorithms are known to require almost $\Omega(n^{1/2})$ queries to test whether an unknown halfspace is monotone versus far from monotone, this shows that adaptivity enables an exponential improvement in the query complexity of monotonicity testing for halfspaces.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases property testing, linear threshold functions, monotonicity, adaptivity

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.38

1 Introduction

Monotonicity testing has been a touchstone problem in property testing for more than fifteen years [17, 23, 19, 22, 21, 3, 1, 24, 29, 6, 8, 27, 10, 11, 12, 7, 14, 25, 13, 4, 16], with many exciting recent developments leading to a greatly improved understanding of the problem in just the past few years. The seminal work of [23] introduced the problem and gave an $O(n/\epsilon)$-query algorithm that tests whether an unknown and arbitrary function $f\colon \{-1,1\}^n \to \{-1,1\}$ is monotone versus ϵ-far from every monotone function. While steady progress followed for non-Boolean functions and for functions over other domains, the first improved algorithm for Boolean-valued functions over $\{-1,1\}^n$ was only achieved in [10], who gave a $\tilde{O}(n^{7/8}) \cdot \mathrm{poly}(1/\epsilon)$-query non-adaptive testing algorithm. A slightly improved $\tilde{O}(n^{5/6}) \cdot \mathrm{poly}(1/\epsilon)$-query non-adaptive algorithm was given by [14], and subsequently [25] gave a $\tilde{O}(n^{1/2}) \cdot \mathrm{poly}(1/\epsilon)$-query non-adaptive algorithm.

On the lower bounds side, the fundamental class of *halfspaces* has played a major role in non-adaptive lower bounds for monotonicity testing to date. We discuss lower bounds for two-sided error monotonicity testing of Boolean-valued functions over $\{-1,1\}^n$, and refer the

* A full version of the paper is available at https://arxiv.org/abs/1706.05556.

† X. C. was supported by NSF grants CCF-1149257 and CCF-1423100. R. A. S. was supported by NSF grants CCF-1420349 and CCF-1563155. L.-Y. T. was supported by NSF grant CCF-1563122. E. W. was supported by the NSF Graduate Research Fellowship under Grant No. DGE-16-44869.

reader to the above references for lower bounds on other variants of the monotonicity testing problem. The first (two-sided) lower bound was established by Fischer et al [22], who used a slight variant of the majority function to give an $\Omega(\log n)$ lower bound for non-adaptive monotonicity testing. More recently, the lower bound of [13], strengthening [14], shows that for any constant $\delta > 0$, there is a constant $\epsilon = \epsilon(\delta) > 0$ such that $\Omega(n^{1/2-\delta})$ non-adaptive queries are required to distinguish whether a Boolean function f – which is promised to be a halfspace – is monotone or ϵ-far from every monotone function. Together with the $\tilde{O}(n^{1/2}) \cdot \mathrm{poly}(1/\epsilon)$-query non-adaptive monotonicity testing algorithm of [25], this shows that halfspaces are "as hard as the hardest functions" to non-adaptively test for monotonicity. Halfspaces are also commonly referred to as "linear threshold functions" or LTFs; for brevity we shall subsequently refer to them as LTFs.

The role of adaptivity

While the above results largely settle the query complexity of non-adaptive monotonicity testing, the situation is less clear when adaptive algorithms are allowed. More generally, the power of adaptivity in property testing is not yet well understood, despite being a natural and important question.[1] A recent breakthrough result of Belovs and Blais [4] gives a $\tilde{\Omega}(n^{1/4})$ lower bound on the query complexity of adaptive algorithms that test whether $f\colon \{-1,1\}^n \to \{-1,1\}$ is monotone versus ϵ-far from monotone, for some absolute constant $\epsilon > 0$. This result was then improved by [16] to $\tilde{\Omega}(n^{1/3})$. [4] also shows that when f is promised to be an "extremely regular" LTF, with regularity parameter at most $O(1)/\sqrt{n}$, then $\log n + O_\epsilon(1)$ adaptive queries suffice. (We define the "regularity" of an LTF in part (a) of Definition 2 below. Here we note only that every n-variable LTF has regularity between $1/\sqrt{n}$ and 1, so $O(1)/\sqrt{n}$-regular LTFs are "extremely regular" LTFs.)

A very compelling question is whether adaptivity helps for monotonicity testing of Boolean functions: can adaptive algorithms go below the [13] $\Omega(n^{1/2-\delta})$-query lower bound for non-adaptive algorithms? While we do not know the answer to this question for general Boolean functions[2], in this work we give a strong positive answer in the case of LTFs, generalizing the upper bound of [4] from "extremely regular" LTFs to arbitrary unrestricted LTFs. The main result of this work is an adaptive algorithm with one-sided error that can test any LTF for monotonicity using $\mathrm{poly}(\log n, 1/\epsilon)$ queries:

▶ **Theorem 1** (Main). *There is a* $\mathrm{poly}(\log n, 1/\epsilon)$*-query*[3] *adaptive algorithm with the following property: given* $\epsilon > 0$ *and black-box access to an unknown LTF* $f\colon \{-1,1\}^n \to \{-1,1\}$*,*

- *If* f *is monotone then the algorithm outputs "monotone" with probability* 1*;*
- *If* f *is* ϵ*-far from every monotone function then the algorithm outputs "non-monotone" with probability at least* $2/3$*.*

---

[1] For monotonicity testing of functions $f\colon [n]^2 \to \{0,1\}$, Berman et al. [5] showed that adaptive algorithms are strictly more powerful than non-adaptive ones (by a factor of $\log 1/\epsilon$). For unateness testing of real-valued functions $f\colon \{0,1\}^n \to \mathbb{R}$, a natural generalization of monotonicity, [2] showed that adaptivity helps by a logarithmic factor. We remark that for another touchstone class in property testing, the class of Boolean juntas, it was only very recently shown [30, 15] that adaptive algorithms are strictly more powerful than non-adaptive algorithms.

[2] For very special functions such as truncated anti-dictators, it is known [22] that adaptive algorithms are known to be much more efficient than nonadaptive algorithms ($O(\log n)$ versus $\Omega(\sqrt{n})$ queries) in finding a violation to monotonicity.

[3] See Theorem 26 of Section 5 for a detailed description of the algorithm's query complexity; we have made no effort to optimize the particular polynomial dependence on $\log n$ and $1/\epsilon$ that the algorithm achieves.

Recalling that the $\Omega(n^{1/2-\delta})$ non-adaptive lower bound from [13] is proved using LTFs as both the yes- and no- functions, Theorem 1 shows that adaptive algorithms are exponentially more powerful than non-adaptive algorithms for testing monotonicity of LTFs. Together with the $\tilde{\Omega}(n^{1/3})$ adaptive lower bound from [16], it also shows that LTFs are exponentially easier to test for monotonicity than general Boolean functions using adaptive algorithms.

1.1 A very high-level overview of the algorithm

The adaptive algorithm of [4] for testing monotonicity of "extremely regular" LTFs is essentially based on a simple binary search over the hypercube $\{-1,1\}^n$ to find an anti-monotone edge[4]. [4] succeeds in analyzing such an algorithm, taking advantage of some of the nice structural properties of regular LTFs, but it is not clear how to carry out such an analysis for general LTFs.

To deal with general LTFs, our algorithm is more involved and employs an iterative stage-wise approach, running for up to $O(\log n)$ stages. Entering the $(t+1)$-th stage, the algorithm maintains a restriction $\rho^{(t)}$ that fixes some of the input variables to f, and in the $(t+1)$-th stage the algorithm queries $f_{\rho^{(t)}}$, where we write $f_{\rho^{(t)}}$ to denote the function f after the restriction $\rho^{(t)}$. At a very high level, in the $(t+1)$-th stage the algorithm either

(i) Obtains definitive evidence (in the form of an anti-monotone edge) that $f_{\rho^{(t)}}$, and hence f, is not monotone. In this case the algorithm halts and outputs "non-monotone." Or, it

(ii) Extends the restriction $\rho^{(t)}$ to obtain $\rho^{(t+1)}$. This is done by fixing a random subset of the variables of expected density $1/2$ that are not fixed under $\rho^{(t)}$, and possibly some additional variables, in such a way as to maintain an invariant described later. Or, it

(iii) Fails to achieve (i) or (ii), which we show is very unlikely to happen. In this case the algorithm simply halts and outputs "monotone."

We describe the invariant of $\rho^{(t)}$ maintained in Case (ii) in Section 1.2. One of its implications in particular is that $f_{\rho^{(t)}}$ is ϵ'-far from monotone, where ϵ' has a polynomial dependence on ϵ. As a result, when the number of surviving variables under $\rho^{(t^*)}$ at the beginning of a stage t^* is at most $\mathrm{poly}(\log n)$, the algorithm can run the simple "edge tester" of [23] on $f_{\rho^{(t^*)}}$ to find an anti-monotone edge with high probability. Although the "edge tester" has query complexity linear in the number of variables, this is affordable since $f_{\rho^{(t^*)}}$ only has $\mathrm{poly}(\log n)$ many variables left. Case (ii) ensures that there are at most $O(\log n)$ stages overall. We will also see that each stage makes at most $\mathrm{poly}(\log n, 1/\epsilon)$ queries; hence the overall query complexity is $\mathrm{poly}(\log n, 1/\epsilon)$.

1.2 A more detailed overview of the algorithm and why it works

In this section we give a more detailed overview of the algorithm and a high-level sketch of its analysis. The algorithm only outputs "non-monotone" if it identifies an anti-monotone edge, so it will correctly output "monotone" on every monotone f with probability 1. Hence, establishing correctness of the algorithm amounts to showing that if f is an LTF that is ϵ-far from monotone, then with high probability the algorithm will output "non-monotone" when it runs on f. Thus, for the remainder of this section, $f(x) = \mathrm{sign}(w_1x_1 + \cdots + w_nx_n - \theta)$ should be viewed as being an LTF that is ϵ-far from monotone.

A crucial notion for understanding the algorithm is that of a (τ, γ, λ)-*non-monotone LTF*.

[4] A *bi-chromatic* edge of $f\colon \{-1,1\}^n \to \{-1,1\}$ is a pair (x,y) of points such that $x, y \in \{-1,1\}^n$ differ at exactly one coordinate and satisfy $f(x) \neq f(y)$. An *anti-monotone* edge of f is a bi-chromatic edge (x,y) that also satisfies $x_i = -1, y_i = 1$ for some $i \in [n]$ and $f(x) = 1, f(y) = -1$.

▶ **Definition 2.** Given an LTF $f\colon \{-1,1\}^S \to \{-1,1\}$ of the form $f(x) = \mathrm{sign}(w \cdot x - \theta)$ over a set of variables S, we say it is a (τ,γ,λ)-*non-monotone LTF with respect to the weights* w if it satisfies the following three properties:

(a) f is τ-*weight-regular*[5] with respect to w, i.e.,

$$\max_{i\in S} |w_i| \le \tau \cdot \sqrt{\sum_{j\in S} w_j^2};$$

(b) f is γ-*balanced*, i.e., $\left|\mathbf{E}_{\mathbf{x}\in\{-1,1\}^n}[f(\mathbf{x})]\right| \le 1-\gamma$; and

(c) f has λ-*significant squared negative weights in* w, i.e.,

$$\frac{\sum_{i\in S: w_i<0}(w_i)^2}{\sum_{i\in S}(w_i)^2} \ge \lambda.$$

Looking ahead, an insight that underlies this definition (as well as our algorithm) is that, when $f = \mathrm{sign}(w\cdot x-\theta)$ is a weight-regular LTF that is far from monotone, f must satisfy (c) above for some large value of λ (see Lemma 12 for a precise formulation). The converse also holds, i.e., an LTF that satisfies all three conditions above must be ϵ-far from monotone for some large value of ϵ (see Lemma 13). This is indeed the reason why we call such functions (τ,γ,λ)-*non-monotone* LTFs. An additional motivation for the regularity condition (a) is that, when f satisfies (c) for some value $\lambda \gg \tau$ (the parameter in (a)), a random restriction ρ (that randomly fixes half of the variables to uniform values from $\{-1,1\}$) would have f_ρ still satisfy (c) with essentially the same λ. The balance condition (b), on the other hand, may be viewed as a technical condition that makes it possible for our various subroutines to work efficiently and correctly; we note that if f is not γ-balanced, then f is trivially $(\gamma/2)$-close to either the monotone function 1 or the monotone function -1.

With Definition 2 in hand, we proceed to a more detailed overview of the algorithm (still at a rather conceptual level). The algorithm takes as input black-box access to $f\colon \{-1,1\}^n \to \{-1,1\}$ and a parameter $\epsilon > 0$. We remind the reader that in the subsequent discussion f should be viewed as an ϵ-far-from-monotone LTF. For the analysis of the algorithm, we also assume that f takes the form of $f(x) = \mathrm{sign}(w_1x_1 + \cdots + w_nx_n - \theta)$, for some unknown (but fixed[6]) weight vector w and threshold θ. They are unknown to the algorithm and will be used in the analysis only.

Our algorithm has two main phases: first an *initialization* phase, and then the phase consisting of the *main procedure*.

Initialization. The algorithm runs an initialization procedure `Regularize-and-Balance`. Roughly speaking, it with high probability either identifies f as a non-monotone LTF by finding an anti-monotone edge and halts, or constructs a restriction $\rho^{(0)}$ such that $f_{\rho^{(0)}}$ becomes a (τ,γ,λ_0)-non-monotone LTF for suitable parameters τ,γ,λ_0, with $\tau = \mathrm{poly}(1/\log n,\epsilon)$, $\gamma = \epsilon$, $\lambda_0 = \mathrm{poly}(\epsilon)$ and $\tau \ll \lambda_0$. In the latter case the algorithm continues with $f_{\rho^{(0)}}$.

Main Procedure. As sketched earlier in Section 1.1 the main procedure operates in a sequence of $O(\log n)$ stages. In its $(t+1)$th stage, it operates on the restricted function

[5] Our terminology "weight-regular" means the same thing as [4]'s "regular." We use the terminology "weight-regular" to distinguish it from the different notion of "Fourier-regularity" which we also require, see Section 2.2.

[6] Note that (w,θ) is not unique for a given f. We pick such a pair and stick to it throughout the analysis.

$f_{\rho^{(t)}}$ which is assumed to be a $(\tau, \gamma, \lambda_t)$-non-monotone LTF, and with high probability either identifies f as non-monotone and halts, or constructs an extension $\rho^{(t+1)}$ of the restriction $\rho^{(t)}$ such that $f_{\rho^{(t+1)}}$ remains $(\tau, \gamma, \lambda_{t+1})$-non-monotone (for some parameter λ_{t+1} that is only slightly smaller than λ_t) while the number of free variables in $\rho^{(t+1)}$ drops by a constant factor.

To describe each stage in more detail, we need the following notation for restrictions. Given a restriction $\rho \in \{-1, 1, *\}^{[n]}$, we use $\textsc{stars}(\rho)$ to denote the set of indices that are not fixed in ρ, i.e., the set of i such that $\rho(i) = *$. Given $f\colon \{-1,1\}^n \to \{-1,1\}$ of the form $f(x) = \text{sign}(\sum w_i x_i - \theta)$, we let $f_\rho\colon \{-1,1\}^{\textsc{stars}(\rho)} \to \{-1,1\}$ denote the function f after the restriction ρ:

$$f_\rho(x) = \text{sign}\left(\textstyle\sum_{i \in \textsc{stars}(\rho)} w_i \cdot x_i + \sum_{j \notin \textsc{stars}(\rho)} w_j \cdot \rho(j) - \theta\right).$$

We stress than the weights of f_ρ remain w_i while the threshold is $\theta - \sum_{j \notin \textsc{stars}(\rho)} w_j \cdot \rho(j)$.

Now for the $(t+1)$th stage, where $t = 0, 1, 2, \ldots$, the main procedure carries out the following sequence of steps (we defer discussion of how these steps are implemented to Section 5). Below for convenience we let g denote $f_{\rho^{(t)}}$, the function that the algorithm operates on in the $(t+1)$th stage.

1. Draw a random subset $A_t \subset \textsc{stars}(\rho^{(t)})$, which consists of roughly half of its variables. Assuming that $\tau \ll \lambda_t$, we have that, with high probability, A_t partitions the positive and negative weights roughly evenly and the collection of weights of variables in $\textsc{stars}(\rho^{(t)}) \setminus A_t$ has λ_{t+1}-significant squared negative weights for some λ_{t+1} that is only slightly smaller than λ_t. (This also justifies the assumption of $\tau \ll \lambda_t$ at the beginning.)
2. Find a restriction $\rho' \in \{-1, 1, *\}^{\textsc{stars}(\rho^{(t)})}$ that fixes the variables in A_t in such a way that $g_{\rho'}$ is 0.96-balanced. The exact constant 0.96 here is not important as long as it is close enough to 1. Note that $g_{\rho'}$ is more balanced than g is promised to be (i.e., $(\gamma = \epsilon)$-balanced and we may assume that $\epsilon \le 0.5$). This helps in the last step of the stage. Our analysis shows that if g is $(\tau, \gamma, \lambda_t)$-non-monotone, then this step succeeds with high probability.
3. Find a set $H_t \subset \textsc{stars}(\rho^{(t)}) \setminus A_t$ that contains those variables x_i that have "high influence" in $g_{\rho'}$. Intuitively, H_t contains variables of $g_{\rho'}$ that violate the τ-weight-regularity condition; after its removal, the collection of weights of variables in $\textsc{stars}(\rho^{(t)}) \setminus (A_t \cup H_t)$ becomes τ-weight-regular again.
4. For each $i \in H_t$, find a bi-chromatic edge of $g_{\rho'}$ on the ith coordinate (this can be done efficiently because the variables in H_t all have high influence in $g_{\rho'}$), which reveals the sign of w_i. If an anti-monotone edge is found, halt and output "non-monotone;" otherwise, we know that the weight of every variable in H_t is positive.
5. Finally, find a restriction $\rho'' \in \{-1, 1, *\}^{\textsc{stars}(\rho^{(t)})}$, which extends ρ' and fixes the variables in $A_t \cup H_t$, such that $g_{\rho''}$ is γ-balanced. Our analysis shows that if g is $(\tau, \gamma, \lambda_t)$-non-monotone and $g_{\rho'}$ is 0.96-balanced, then this step succeeds with high probability. By Step 3, $g_{\rho''}$ is τ-weight-regular. In addition, $g_{\rho''}$ has λ_{t+1}-significant squared negative weights because of Step 1 and Step 4 (which makes sure that all variables in H_t have positive weights). At the end, we set $\rho^{(t+1)}$ to be the composition of $\rho^{(t)}$ and ρ'' and move on to the next stage.

To summarize, our analysis shows that if $f_{\rho^{(t)}}$ is $(\tau, \gamma, \lambda_t)$-non-monotone (entering the $(t+1)$th stage) then with high probability the algorithm in the $(t+1)$th stage either finds an anti-monotone edge and halts, or finds an extension $\rho^{(t+1)}$ of $\rho^{(t)}$ such that:

(i) The new function $f_{\rho^{(t+1)}}$ is $(\tau, \gamma, \lambda_{t+1})$-non-monotone (entering the $(t+2)$th stage), where the parameter λ_{t+1} is only slightly smaller than λ_t (more on this below); and
(ii) The number of surviving variables in $\rho^{(t+1)}$ is only about half of that of $\rho^{(t)}$.

This implies that, with high probability, the main procedure within $O(\log n)$ stages either finds an anti-monotone edge and returns the correct answer "non-monotone" or constructs a restriction $\rho^{(t)}$ such that $f_{\rho^{(t)}}$ is $(\tau, \gamma, \lambda_t)$-non-monotone and the number of surviving variables under $\rho^{(t)}$ is at most $m = \mathrm{poly}(\log n, 1/\epsilon)$. For the latter case, our analysis (Lemma 13) together with the fact that λ_t drops only slightly in each stage show that $f_{\rho^{(t)}}$ remains $\epsilon' = \mathrm{poly}(\epsilon)$-far from monotone. Thus, the algorithm concludes by running the "edge tester" from [23] to ϵ'-test the m-variable function $f_{\rho^{(t)}}$, which uses $O(m/\epsilon') = \mathrm{poly}(\log n, 1/\epsilon)$ queries to $f_{\rho^{(t)}}$ and finds an anti-monotone edge with high probability. To summarize, when f is an LTF that is ϵ-far from monotone, our algorithm finds an anti-monotone edge and outputs "non-monotone" with high probability. As discussed earlier at the beginning of Section 1.2 about its one-sideness, the correctness of the algorithm follows.

1.3 Relation to previous work

We have already discussed how our main result, Theorem 1, relates to the recent upper and lower bounds of [25, 13, 4] for monotonicity testing. At the level of techniques, several aspects of our algorithm are reminiscent of some earlier work in property testing of Boolean functions and probability distributions as we describe below.

At a high level, the $\mathrm{poly}(1/\epsilon)$-query algorithm of [26] for testing whether a function is an LTF identifies high-influence variables and "deals with them separately" from other variables, as does our algorithm. The more recent algorithm of [28], for testing whether a function is a signed majority function, like our algorithm proceeds in a series of stages which successively builds up a restriction by fixing more and more variables. Like our algorithm the [28] algorithm makes only $\mathrm{poly}(\log n, 1/\epsilon)$ adaptive queries, but there are many differences both between the two algorithms and between their analyses. To briefly note a few of these differences, the [28] algorithm has two-sided error while our algorithm has one-sided error; the former also heavily leverages both the very "rigid" structure of the degree-1 Fourier coefficients of any signed majority function and the near-perfect balancedness of any signed majority function between the two outputs 1 and -1, neither of which hold in our setting. Finally, we note that the general approach of iteratively selecting and retaining a random subset of the remaining "live" elements, then doing some additional pruning to identify, check, and discard a small number of "heavy" elements, then proceeding to the next stage is reminiscent of the Approx-Eval-Simulator procedure of [9], which deals with testing probability distributions in the "conditional sampling" model.

1.4 Organization

In Section 2 we recall the necessary background concerning monotonicity, LTFs, and restrictions, and state a few useful algorithmic and structural results from prior work. In Section 3 we establish several new structural results about "regular" LTFs: we first show that its distance to monotonicity corresponds (approximately) to its total amount of squared negative coefficient weights; we also prove that its distance to monotonicity is preserved under a random restriction to a set of its non-decreasing variables. In Section 4 we present and analyze some simple algorithmic subroutines that will be used to identify high influence variables and check that they are non-decreasing. Finally in Section 5, we give a detailed description of our

overall algorithm for testing monotonicity of LTFs, and prove its correctness, establishing our main result (Theorem 1).

2 Background

We write $[n]$ for $\{1, \dots, n\}$, and use boldface letters (e.g. $\mathbf{x}$ and $\mathbf{X}$) to denote random variables. We briefly recall some basic notions. A function $f\colon \{-1,1\}^n \to \{-1,1\}$ is *monotone* (short for "monotone non-decreasing") if $x \preceq y$ implies $f(x) \leq f(y)$, where "$x \preceq y$" means that $x_i \leq y_i$ for all $i \in [n]$. A function f is *unate* if there is a bit vector $a \in \{-1,1\}^n$ such that $f(a_1 x_1, \dots, a_n x_n)$ is monotone. It is well known that every LTF (defined below) is unate.

We measure distance between functions $f, g\colon \{-1,1\}^n \to \{-1,1\}$ with respect to the uniform distribution, so we say that f and g are ϵ-*close* if

$$\mathrm{dist}(f,g) \colon = \Pr_{\mathbf{x}\in\{-1,1\}^n}\left[f(\mathbf{x}) \neq g(\mathbf{x})\right] \leq \epsilon,$$

and that f and g are ϵ-*far* otherwise. A function f is ϵ-*far from monotone* if it is ϵ-far from every monotone function g. We write $\mathrm{dist}(f, \textsc{Mono})$ to denote the minimum value of $\mathrm{dist}(f,g)$ over all monotone functions g. Throughout the paper all probabilities and expectations are with respect to the uniform distribution over $\{-1,1\}^n$ unless otherwise indicated. As indicated in Definition 2, we say that a $\{-1,1\}$-valued function f is γ-*balanced* if

$$\left|\mathop{\mathbf{E}}_{\mathbf{x}\in\{-1,1\}^n}[f(\mathbf{x})]\right| \leq 1-\gamma.$$

A function $g\colon \{-1,1\}^n \to \{-1,1\}$ is a *junta* over $S \subseteq [n]$ if g depends only on the coordinates in S. We say f is ϵ-*close* to a *junta* over S if f is ϵ-close to g for some g that is a junta over S.

2.1 LTFs and weight-regularity

A function $f\colon \{-1,1\}^n \to \{-1,1\}$ is an *LTF* (also commonly referred to as a *halfspace*) if there exist real weights $w_1, \dots, w_n \in \mathbb{R}$ and a real threshold $\theta \in \mathbb{R}$ such that

$$f(x) = \begin{cases} 1 & \text{if } w_1 x_1 + \cdots + w_n x_n \geq \theta, \\ -1 & \text{if } w_1 x_1 + \cdots + w_n x_n < \theta. \end{cases}$$

We say that $w = (w_1, \dots, w_n)$ are the *weights* and θ the *threshold* of the LTF, and we say that (w, θ) *represents* the LTF f, or simply that $f(x)$ is the LTF given by $\mathrm{sign}(w \cdot x - \theta)$. Note that for any LTF f there are in fact infinitely many pairs (w, θ) that represent f; we fix a particular pair (w, θ) for each n-variable LTF f and work with it in what follows.

An important notion in our arguments is that of *weight-regularity*. As indicated in Definition 2, given a weight vector $w \in \mathbb{R}^n$, we say that w is τ-*weight-regular* if no more than a τ-fraction of the 2-norm of $w = (w_1, \dots, w_n)$ comes from any single coefficient w_i, i.e.,

$$\max_{i\in[n]} |w_i| \leq \tau \cdot \sqrt{w_1^2 + \cdots + w_n^2}. \tag{1}$$

If we have fixed a representation (w, θ) for f such that w is τ-weight-regular, we frequently abuse the terminology and say that f is τ-weight-regular.

2.2 Fourier analysis of Boolean functions and Fourier-regularity

Given a function $f\colon \{-1,1\}^n \to \mathbb{R}$, we define its *Fourier coefficients* by $\hat{f}(S) = \mathbf{E}[f \cdot x_S]$ for each $S \subseteq [n]$, where x_S denotes $\prod_{i\in S} x_i$, and we have that $f(x) = \sum_S \hat{f}(S) \cdot x_S$. We will be particularly interested in f's *degree*-1 coefficients, i.e., $\hat{f}(S)$ for $|S| = 1$; we will write these as $\hat{f}(i)$ rather than $\hat{f}(\{i\})$. We recall *Plancherel's identity* $\langle f, g\rangle = \sum_S \hat{f}(S)\hat{g}(S)$, which has as a special case *Parseval's identity*, $\mathbf{E}_{\mathbf{x}}[f(\mathbf{x})^2] = \sum_S \hat{f}(S)^2$. It follows that every $f\colon \{-1,1\}^n \to \{-1,1\}$ has $\sum_S \hat{f}(S)^2 = 1$.

We further recall that, for any unate function $f\colon \{-1,1\}^n \to \{-1,1\}$ (and hence any LTF), we have $|\hat{f}(i)| = \mathbf{Inf}_i(f)$, where the *influence* of variable i on f is

$$\mathbf{Inf}_i(f) = \Pr_{\mathbf{x}\in\{-1,1\}^n}\big[f(\mathbf{x}) \neq f(\mathbf{x}^{\oplus i})\big],$$

where $x^{\oplus i}$ is the vector obtained from x by flipping coordinate i.

We say that $f\colon \{-1,1\}^n \to \{-1,1\}$ is τ-*Fourier-regular* if $\max_{i\in[n]} |\hat{f}(i)| \leq \tau$. Section 2.5 summarizes some relationships between weight-regularity and Fourier-regularity of LTFs.

2.3 Restrictions

A *restriction* ρ is an element of $\{-1,1,*\}^{[n]}$; we view ρ as a partial assignment to the n variables $x_1,\dots,x_n$, where $\rho(i) = *$ indicates that variable x_i is unassigned. We write $\mathrm{supp}(\rho)$ to denote the set of indices i such that $\rho(i) \in \{-1,1\}$ and $\textsc{stars}(\rho)$ to denote the set of i such that $\rho(i) = *$ (and thus, $\textsc{stars}(\rho)$ is the complement of $\mathrm{supp}(\rho)$).

Given restrictions $\rho, \rho' \in \{-1,1,*\}^{[n]}$ we say that ρ' is an *extension* of ρ if $\mathrm{supp}(\rho) \subseteq \mathrm{supp}(\rho')$ and $\rho'(i) = \rho(i)$ for all $i \in \mathrm{supp}(\rho)$. If ρ and ρ' are restrictions with disjoint support we write $\rho\rho'$ to denote the *composition* of these two restrictions (that has support $\mathrm{supp}(\rho) \cup \mathrm{supp}(\rho')$).

2.4 Useful algorithmic tools from prior work

We recall some algorithmic tools for working with black-box functions $f\colon \{-1,1\}^n \to \{-1,1\}$.

Estimating sums of squares of degree-1 Fourier coefficients. We first recall Corollary 16 of [26] (slightly specialized to our context):

▶ **Lemma 3** (Corollary 16 [26]). *There is a procedure* `Estimate-Sum-of-Squares`(f, T, η, δ) *with the following properties. Given as input black-box access to* $f\colon \{-1,1\}^n \to \{-1,1\}$*, a subset* $T \subseteq [n]$*, and parameters* $\eta, \delta > 0$*, it runs in time* $O(n \cdot \log(1/\delta)/\eta^4)$*, makes* $O(\log(1/\delta)/\eta^4)$ *queries, and with probability at least* $1-\delta$ *outputs an estimate of* $\sum_{i\in T} \hat{f}(i)^2$ *that is accurate to within an additive* $\pm\eta$*.*

Checking Fourier regularity. We recall Lemma 18 of [26], which is an easy consequence of Lemma 3:

▶ **Lemma 4** (Lemma 18 [26]). *There is a procedure* `Check-Fourier-Regular`(f, T, τ, δ) *with the following properties. Given as input black-box access to* $f\colon \{-1,1\}^n \to \{-1,1\}$*,* $T \subseteq [n]$*, and* $\tau, \delta > 0$*, it runs in time* $O(n \cdot \log(1/\delta)/\tau^{16})$*, makes* $O(\log(1/\delta)/\tau^{16})$ *queries, and*

- *If* $|\hat{f}(i)| \geq \tau$ *for some* $i \in T$ *then it outputs "not regular" with probability* $1-\delta$*;*
- *If every* $i \in T$ *has* $|\hat{f}(i)| \leq \tau^2/4$ *then it outputs "regular" with probability* $1-\delta$*.*

Estimating the mean. For completeness we recall the following simple fact (which follows from a standard Chernoff bound):

▶ **Fact 5.** *There is a procedure* `Estimate-Mean`(f, ϵ, δ) *with the following properties. Given as input black-box access to* $f\colon \{-1,1\}^n \to \{-1,1\}$ *and* $\epsilon, \delta > 0$*, it makes* $O(\log(1/\delta)/\epsilon^2)$ *queries and with probability at least* $1-\delta$ *it outputs a value* $\tilde{\mu}$ *such that* $|\tilde{\mu} - \mu| \leq \epsilon$*, where* $\mu = \mathbf{E}_{\mathbf{x}\in\{-1,1\}^n}[f(\mathbf{x})]$.

The edge tester of [23]. We recall the performance guarantee of the "edge tester" (which works by querying both endpoints of uniform random edges and outputting "non-monotone" if and only if it encounters an anti-monotone edge):

▶ **Theorem 6** ([23]). *There is a procedure* `Edge-Tester`(f, ϵ, δ) *with the following properties: Given black-box access to* $f\colon \{-1,1\}^n \to \{-1,1\}$ *and parameters* $\epsilon, \delta > 0$*, it makes* $O(n\log(1/\delta)/\epsilon)$ *queries and outputs either "monotone" or "non-monotone" such that:*

- *If* f *is monotone then it outputs "monotone" with probability 1;*
- *If* f *is* ϵ*-far from monotone then it outputs "non-monotone" with probability at least* $1-\delta$.

2.5 Useful structural results from prior work

Gaussian distributions and the Berry–Esséen theorem. Recall that the p.d.f. of the standard Gaussian distribution $\mathcal{N}(0,1)$ with mean 0 and variance 1 is given by

$$\phi(x) = \frac{1}{\sqrt{2\pi}} \cdot e^{-x^2/2}.$$

The Berry–Esséen theorem (see e.g., [20]) is a version of the central limit theorem for sums of independent random variables (stating that such a sum converges to a normal distribution) that provides a quantitative error bound. It is useful for analyzing weight-regular LTFs and we recall it below (as well as the standard Hoeffding inequality).

▶ **Theorem 7** (Berry–Esséen). *Let* $\ell(\mathbf{x}) = c_1\mathbf{x}_1 + \cdots + c_n\mathbf{x}_n$ *be a linear form of* n *unbiased, independent random* $\{\pm 1\}$*-valued variables* $\mathbf{x}_i$*. Let* τ *be such that* $|c_i| \leq \tau$ *for all* i*, and let* $\sigma = (\sum c_i^2)^{1/2}$*. Write* F *for the c.d.f. of* $\ell(\mathbf{x})/\sigma$*, i.e.,* $F(t) = \Pr[\ell(\mathbf{x})/\sigma \leq t]$*. Then for all* $t \in \mathbb{R}$*, we have that* $|F(t) - \Phi(t)| \leq \tau/\sigma$*, where* Φ *denotes the c.d.f. of a standard* $\mathcal{N}(0,1)$ *Gaussian random variable.*

▶ **Theorem 8** (Hoeffding's Inequality). *Let* $\mathbf{x}$ *be a random variable drawn uniformly from* $\{-1,1\}^n$*. Let* $w \in \mathbb{R}^d$ *and* $t > 0$*. Then we have*

$$\Pr_{\mathbf{x}}\left[|\mathbf{x}\cdot w| \geq t\right] \leq 2\exp\left(-\frac{t^2}{2\|w\|_2^2}\right) \quad \text{and} \quad \Pr_{\mathbf{x}}\left[\mathbf{x}\cdot w \geq t\right] \leq \exp\left(-\frac{t^2}{2\|w\|_2^2}\right).$$

Weight-regularity versus Fourier-regularity for LTFs. An easy argument, using the Berry–Esséen, shows that weight-regularity always implies Fourier-regularity for LTFs:

▶ **Theorem 9** (Theorem 38 of [26]). *Let* $f\colon \{-1,1\}^n \to \{-1,1\}$ *be a* τ*-weight-regular LTF. Then* f *is* $O(\tau)$*-Fourier-regular.*

The converse is not always true; for example, the constant 1 function, which is τ-Fourier-regular for all $\tau > 0$, may be written as $f(x) = \mathrm{sign}(x_1 + 2)$. However, if we additionally impose the condition that f is not too biased towards $+1$ or -1, then a converse holds. Sharpening an earlier result (Theorem 39 of [26]), Dzindzalieta has proved the following:

▶ **Theorem 10** (Theorem 20 of [18]). *Let* $f(x) = \mathrm{sign}(w\cdot x - \theta)$ *be an LTF such that* $|\mathbf{E}_{\mathbf{x}}[f(\mathbf{x})]| \leq 1-\gamma$*. If* f *is* τ*-Fourier-regular, then it is also* $O(\tau/\gamma)$*-weight-regular.*

Making LTFs Fourier-regular by fixing high-influence variables. Finally, we will need the following simple result (Proposition 62 from [26]), which shows that LTFs typically become Fourier-regular when their highest-influence variables are fixed to constants:

▶ **Proposition 11.** *Let $f\colon \{-1,1\}^n \to \{-1,1\}$ be an LTF and let $J \supseteq \{i : |\hat{f}(i)| \geq \beta\}$. Then f_ρ is not (β/η)-Fourier-regular for at most an η-fraction of all $2^{|J|}$ restrictions ρ that fix variables in J.*

3 New structural results about LTFs

Our analysis requires a few new structural results about LTFs. We collect these results in this section; their proofs can be found in the full version.

First we show that, for weight-regular LTFs, the distance to monotonicity corresponds (approximately) to its total amount of squared weights of negative coefficients (under any representation (w,θ)). Lemma 12 below shows that if f is far from monotone then this quantity is large, and Lemma 13 establishes a converse (both for weight-regular LTFs). We note that Lemma 12 is essentially equivalent to a lemma proved in [4].

We introduce some notation. Given an LTF $f\colon \{-1,1\}^n \to \{-1,1\}$ with $f(x) = \mathrm{sign}(w \cdot x - \theta)$, we let $P = P(f)$ and $N = N(f)$ denote the set of non-negative and negative indices, respectively: $P = \{i \in [n]\colon w_i \geq 0\}$ and $N = \{j \in [n]\colon w_j < 0\}$. We let $\mathrm{pos}(f)$ and $\mathrm{neg}(f)$ denote the sum of squared weights of positive and negative coefficients, respectively:

$$\mathrm{pos}(f) = \sum_{i \in P} w_i^2 \quad \text{and} \quad \mathrm{neg}(f) = \sum_{j \in N} w_j^2.$$

Recall that we say f has λ-*significant squared negative weights* if $\mathrm{neg}(f)/(\mathrm{pos}(f)+\mathrm{neg}(f)) \geq \lambda$.

We state Lemma 12 and Lemma 13. Their proofs can be found in the full version.

▶ **Lemma 12.** *Let $f\colon \{-1,1\}^n \to \{-1,1\}$ be an LTF given by $f(x) = \mathrm{sign}(w \cdot x - \theta)$. If f is both ϵ-far from monotone and τ-weight-regular for some $\tau \leq \epsilon/16$, then f must have λ-significant squared negative weights, where $\lambda = \epsilon^2/(16\ln(8/\epsilon))$.*

▶ **Lemma 13.** *Let $f(x) = \mathrm{sign}(\sum_{i\in[n]} w_i \cdot x_i - \theta)$ be (τ,γ,λ)-non-monotone with $\tau \leq \sqrt{\lambda}/16$. Then we have*

$$\mathrm{dist}(f, \textsc{Mono}) \geq \min\left\{\Omega\big(\sqrt{\lambda}\gamma^2\big) - O(\tau), \Omega\left(\frac{\gamma^3}{\ln(8/\gamma)}\right) - O(\tau\gamma)\right\}.$$

Our next goal is to show that for any LTF $f\colon \{-1,1\}^n \to \{-1,1\}$, a random restriction that fixes variables of f that are monotonically non-decreasing has, in expectation, the same distance to monotonicity as the original function f. We state Lemma 14 below, which will be used later in the proof of Lemma 19. Its proof can be found in the full version.

▶ **Lemma 14.** *Let $f\colon \{-1,1\}^n \to \{-1,1\}$ be an LTF and let $S \subseteq [n]$ be a set of variables of f that are monotonically non-decreasing. Then a random restriction $\boldsymbol{\rho}$ that fixes each variable in S independently and uniformly to a random element of $\{-1,1\}$ satisfies*

$$\mathop{\mathbf{E}}_{\boldsymbol{\rho}}\Big[\mathrm{dist}(f_{\boldsymbol{\rho}}, \textsc{Mono})\Big] = \mathrm{dist}(f, \textsc{Mono}).$$

4 Algorithmic tools for LTFs

Our algorithm uses a few simple subroutines that may be viewed as relatively low-level algorithmic tools for working with LTFs. We present these tools in this section; the underlying algorithms and their analysis can be found in the full version.

We start with a subroutine `Find-Hi-Influence-Vars` that finds high-influence variables.

▶ **Lemma 15.** *Suppose that the subroutine* `Find-Hi-Influence-Vars`(f, ρ, τ, δ) *is called on a function* $f\colon \{-1,1\}^n \to \{-1,1\}$*, a restriction* $\rho \in \{-1,1,*\}^n$*, and parameters* $\tau, \delta > 0$*. Then it runs in* $\tilde{O}(\log n \cdot \log(1/\delta)/\tau^{10}) \cdot n$ *time, makes at most* $\tilde{O}(\log n \cdot \log(1/\delta)/\tau^{10})$ *queries, and with probability at least* $1-\delta$ *it outputs a set* $H \subseteq \text{STARS}(\rho)$ *such that:*

- *If* $|\widehat{f}_\rho(i)| \geq \tau$ *then* $i \in H$*;*
- *If* $|\widehat{f}_\rho(i)| < \tau/2$ *then* $i \notin H$*.*

Given an LTF, the next subroutine `Check-Weight-Positive` checks whether the weight of a variable is positive.

▶ **Lemma 16.** *Suppose that the subroutine* `Check-Weight-Positive`$(f, \rho, i, \tau, \delta)$ *is called on an LTF* $f(x) = \text{sign}(\sum_{i=1}^n w_i x_i - \theta)$*, a restriction* $\rho \in \{-1,1,*\}^n$*,* $i \in \text{STARS}(\rho)$*, and two parameters* $\tau, \delta > 0$ *such that* $|\widehat{f}_\rho(i)| \geq \tau$ *(note that the latter implies that* $w_i \neq 0$*). Then it runs in* $O(\log(1/\delta)/\tau) \cdot n$ *time, makes* $O(\log(1/\delta)/\tau)$ *queries, and:*

- *If it does not output "fail", which happens with probability at most* δ*;*
- *It outputs "positive" if* $w_i > 0$*, and it outputs "negative" if* $w_i < 0$*.*

5 Detailed description of the algorithm

We present our algorithm and its analysis in this section.

5.1 The algorithm

Our main testing algorithm, `Mono-Test-LTF`, is presented in Figure 1. Its main components are two procedures called `Regularize-and-Balance` and `Main-Procedure`, described and analyzed in Sections 5.2 and 5.3. As will become clear later, `Mono-Test-LTF` is one-sided, i.e., it always outputs "monotone" when the input function f is monotone (because it only outputs "non-monotone" when an anti-monotone edge is found, via `Check-Weight-Positive` or `Edge-Tester`). Thus, our analysis of correctness below focuses on the case when f is an LTF that is ϵ-far from monotone, and shows that in this case `Mono-Test-LTF` outputs "non-monotone" with probability at least $2/3$.

5.2 Key properties of procedure `Regularize-and-Balance`

Let $f\colon \{-1,1\}^n \to \{-1,1\}$ be an LTF, given by $f(x) = \text{sign}(w \cdot x - \theta)$. Assume that f is ϵ-far from monotone. The goal of the procedure `Regularize-and-Balance`(f, ϵ) is to return a restriction $\rho \in \{-1,1,*\}^{[n]}$ such that f_ρ is a $(\tau, \epsilon, \lambda)$-non-monotone LTF (with respect to (w, θ)), where

$$\lambda = \frac{\epsilon^2}{36 \ln(12/\epsilon)} \quad \text{and} \quad \tau = \frac{\lambda \epsilon}{\log^2 n}. \tag{2}$$

Here is some intuition that may be helpful in understanding `Regularize-and-Balance`. If the procedure halts and outputs "monotone" in Step 2, this signals that the (low-probability) failure event of `Find-Hi-Influence-Variables` has taken place (since it has

Algorithm `Mono-Test-LTF`(f, ϵ)
Input: Oracle access to an LTF $f\colon \{-1,1\}^n \to \{-1,1\}$ and a parameter $\epsilon > 0$.
Output: Returns "monotone" or "non-monotone."
1. Call `Regularize-and-Balance`(f, ϵ). If it returns a restriction $\rho \in \{-1, 1, *\}^{[n]}$ then continue to Step 2; if it returns "non-monotone," halt and output "non-monotone;" if it returns "monotone," halt and output "monotone."
2. Call `Main-Procedure`(f, ρ, ϵ). If it returns "non-monotone," halt and output "non-monotone;" if it returns "monotone," halt and output "monotone."

Figure 1 Main algorithm `Mono-Test-LTF`. If f is monotone it outputs "monotone" with probability 1; if f is ϵ-far from monotone, it outputs "non-monotone" with probability $\geq 2/3$.

spuriously identified more variables as having high influence than is possible given Parseval's identity; see Lemma 15). The procedure halts and outputs "non-monotone" in Step 3 only if `Check-Weight-Positive` has unambiguously found an anti-monotone edge. If the procedure outputs "monotone" in Step 3, this signals the (low-probability) event that `Check-Weight-Positive` failed to identify some index $i \in H$ (which was supposed to have high influence) as either having $w_i > 0$ or $w_i < 0$. Finally if it outputs "monotone" in Step 4, this signals that f appears to be close to monotone.[7]

It is clear that `Regularize-and-Balance` is one-sided.

▶ **Fact 17.** *`Regularize-and-Balance`(f, ϵ) never returns "non-monotone" if f is monotone.*

We also have the following upper bound for the number of queries it uses (which can be straight forwardly verified by tracing through procedure calls and parameter settings):

▶ **Fact 18.** *The number of queries used by `Regularize-and-Balance`(f, ϵ) is $\tilde{O}(\log^{41} n/\epsilon^{90})$.*

We prove the main property of the procedure `Regularize-and-Balance` in Appendix A.

▶ **Lemma 19.** *If $f(x) = \mathrm{sign}(w \cdot x - \theta)$ is ϵ-far from monotone, then with probability at least $9/10$, `Regularize-and-Balance`(f, ϵ) returns either "non-monotone," or a restriction ρ such that f_ρ is a $(\tau, \epsilon, \lambda)$-non-monotone LTF with respect to (w, θ).*

5.3 Key properties of `Main-Procedure`

`Main-Procedure` is presented in Figure 3. Given Lemma 19 we may assume that the input (f, ρ, ϵ) satisfies that f_ρ is a $(\tau, \epsilon, \lambda)$-non-monotone LTF (see the choices of τ and λ in (2)).

We prove the following main lemma in this section.

▶ **Lemma 20.** *`Main-Procedure`(f, ρ, ϵ) never returns "non-monotone" when f is monotone. When f_ρ is a $(\tau, \epsilon, \lambda)$-non-monotone LTF, it returns "non-monotone" with probability at least $81/100$.*

The procedure only returns "non-monotone" when it finds an anti-monotone edge in the subroutine `Check-Weight-Positive`. Hence we may focus on the case when f_ρ is $(\tau, \epsilon, \lambda)$-non-monotone. For this purpose, we analyze the three steps 2(a), 2(b), 2(c) of each while loop of `Main-Procedure`, and prove the following lemma.

[7] This will become clear later in the proof of Lemma 19 where we show that Step 4 fails with low probability when f is far from monotone.

Procedure `Regularize-and-Balance`(f, ϵ)
Input: Parameter $\epsilon > 0$ and black-box oracle access to an LTF $f\colon \{-1,1\}^n \to \{-1,1\}$ of the form $f(x) = \mathrm{sign}(w \cdot x - \theta)$, with unknown weights w and threshold θ.
Output: Either "non-monotone," "monotone," or a restriction $\rho \in \{-1,1,*\}^{[n]}$.

1. Let $C_{RB} > 0$ be a large enough constant; let τ' and δ be the following parameters:
$$\tau' = \tau^2\epsilon^3/C_{RB} \quad \text{and} \quad \delta = \tau'^2/C_{RB}.$$
2. Call `Find-Hi-Influence-Vars`$(f, (*)^n, \tau', \delta)$ and let H be the set it returns. If $|H| > 4/\tau'^2$, halt and output "monotone."
3. For each $i \in H$, call `Check-Weight-Positive`$(f, (*)^n, i, \tau'/2, \delta)$. If any call returns "negative," halt and output "non-monotone;" if any call returns "fail," halt and output "monotone;" otherwise (when all calls return "positive") continue to Step 4.
4. Repeat C_{RB}/ϵ times:
   Draw a restriction ρ, which has support H and is obtained by selecting a random assignment from $\{-1,1\}^H$. Call
   `Check-Fourier-Regular`$(f_\rho, [n] \setminus H, \sqrt{12\tau'/\epsilon}, \delta/2)$
   and `Estimate-Mean`$(f_\rho, \epsilon/6, \delta/2)$.
   Halt and output the first ρ where `Check-Fourier-Regular` outputs "regular" and `Estimate-Mean` returns a number of absolute value $\leq 1 - 7\epsilon/6$. If the procedure fails to find such a restriction ρ, halt and output "monotone."

Figure 2 Procedure `Regularize-and-Balance`. Our analysis (Lemma 19) focuses on the case when f is ϵ-far from monotone.

▶ **Lemma 21.** *Let $t \leq 4\log n$, and suppose that at the beginning of the $(t+1)$th loop of `Main-Procedure`, $f_{\rho^{(t)}}$ is $(\tau, \epsilon, \lambda(1 - t/(8\log n)))$-non-monotone. Then with probability at least $1 - 1/(40\log n)$, it either returns "non-monotone" within this loop or obtains a set $A_t \subseteq [n] \setminus \mathrm{supp}(\rho^{(t)})$ and a restriction $\rho^{(t+1)}$ extending $\rho^{(t)}$ at the end of this loop such that*
1. $|A_t| \geq |\textsc{stars}(\rho^{(t)})|/4$;
2. $\mathrm{supp}(\rho^{(t)}) \cup A_t \subseteq \mathrm{supp}(\rho^{(t+1)})$; *and*
3. $f_{\rho^{(t+1)}}$ *is a* $(\tau, \epsilon, \lambda(1 - (t+1)/(8\log n)))$*-non-monotone LTF.*

We use Lemma 21 to prove Lemma 20 in Appendix B.1.

5.3.1 Proof of Lemma 21

The proof of Lemma 21 consists of three lemmas, one for each steps 2(a), 2(b) and 2(c). Below we assume that the condition of Lemma 21 holds at the beginning of the $(t+1)^{\text{th}}$ loop, for some $t \leq 4\log n$. We introduce the following notation for convenience. We let $I = \textsc{stars}(\rho^{(t)})$, with $m = |I|$. Given the random subset A_t of I found in Step 2(a), we let $B_t = I \setminus A_t$. Also note that $m \geq 1/\tau^2$.

We start with the lemma for Step 2(a), which states that with high probability, A_t is large and splits the weights (both positive and negative) in I evenly. We present the proof in Appendix B.2.

Procedure `Main-Procedure`(f, ρ, ϵ)
Input: Parameter $\epsilon > 0$, oracle access to an LTF $f\colon \{-1,1\}^n \to \{-1,1\}$ of the form $f(x) = \mathrm{sign}(w \cdot x - \theta)$ with unknown weights w and threshold θ, and a restriction ρ.
Output: Either "non-monotone" or "monotone."

1. Set $t = 0$ and $\rho^{(0)} = \rho$.
2. While $|\textsc{stars}(\rho^{(t)})| \geq 1/\tau^2$, repeat the following steps:
   a. Construct a subset $A_t \subseteq \textsc{stars}(\rho^{(t)})$ by independently putting each index $i \in \textsc{stars}(\rho^{(t)})$ into A_t with probability $1/2$.
   b. Call `Find-Balanced-Restriction`$(f, \rho^{(t)}, A_t, \epsilon)$. If it returns "monotone" then halt and return "monotone;" otherwise, it returns a restriction ρ' with $\mathrm{supp}(\rho') = \mathrm{supp}(\rho^{(t)}) \cup A_t$.
   c. Call `Maintain-Regular-and-Balance`(f, ρ', ϵ). If it returns "non-monotone" then halt and output "non-monotone;" if it returns "monotone" then halt and output "monotone;" otherwise, it returns a restriction η and we set $\rho^{(t+1)}$ to $\rho'\eta$.
   d. Increment t by 1. If $t > 4 \log n$, halt and output "monotone;" otherwise proceed to the next iteration of step (a) of the loop.
3. Let $\epsilon' = \epsilon^3/(C \log(1/\epsilon))$ for some large constant C; run `Edge-Tester`$(f_{\rho^{(t)}}, \epsilon', 1/10)$ and output what it outputs (either "monotone" or "non-monotone").

Figure 3 Procedure `Main-Procedure`. Our analysis in Section 5.3 focuses on the case when f_ρ is a $(\tau, \epsilon, \lambda)$-non-monotone LTF.

Subroutine `Find-Balanced-Restriction`$(f, \rho^{(t)}, A_t, \epsilon)$
Input: Access to $f\colon \{-1,1\}^n \to \{-1,1\}$, restriction $\rho^{(t)}$, $A_t \subseteq \textsc{stars}(\rho^{(t)})$, and $\epsilon > 0$.
Output: "monotone" or a ρ' with $\mathrm{supp}(\rho') = \mathrm{supp}(\rho^{(t)}) \cup A_t$ that extends $\rho^{(t)}$.

Repeat $C_{BR} \cdot \log n/\epsilon^3$ times for some large enough constant C_{BR}:

Draw a ρ^*, which has support A_t and is obtained by selecting a random assignment from $\{-1,1\}^{A_t}$, and let $\rho' = \rho^{(t)}\rho^*$. Call `Estimate-Mean`$(f_{\rho'}, 0.01, \delta)$, where $\delta = \epsilon^3/(200 C_{BR} \log^2 n)$. If it returns a number of absolute value at most 0.03, halt and output ρ'.

Otherwise, output "monotone."

Figure 4 Subroutine `Find-Balanced-Restriction`. We are interested in the case when $f_{\rho^{(t)}}$ is a $(\tau, \epsilon, \lambda(1 - t/(8 \log n)))$-non-monotone LTF, and A_t satisfies the conditions of Lemma 23.

▶ **Lemma 22.** *Assume that $f_{\rho^{(t)}}$ is a $(\tau, \epsilon, \lambda(1 - t/(8 \log n))$-non-monotone LTF. With probability at least $1 - \exp(-\Omega(\log^2 n))$, A_t and B_t satisfy $|A_t| \geq m/4$,*

$$\frac{1}{2} - \frac{1}{32 \log n} \leq \frac{\sum_{i \in A_t} w_i^2}{\sum_{i \in I} w_i^2} \leq \frac{1}{2} + \frac{1}{32 \log n} \quad \text{and} \quad \frac{\sum_{i \in B_t : w_i < 0} w_i^2}{\sum_{i \in B_t} w_i^2} \geq \lambda \left(1 - \frac{t+1}{8 \log n}\right). \quad (3)$$

We give `Find-Balanced-Restriction` in Figure 4 and show the following lemma for Step 2(b). (The `Find-Balanced-Restriction` subroutine is similar to Algorithm 1 of [28], and Lemma 23 and its proof (presented in Appendix B.3) are reminiscent of Lemma 7 of [28]; however, because of some technical differences we cannot directly apply those results, so we give a self-contained presentation here.)

Subroutine `Maintain-Regular-and-Balanced`(f, ρ', ϵ)
Input: Oracle access to $f\colon \{-1,1\}^n \to \{-1,1\}$, restriction ρ', parameter $\epsilon > 0$.
Output: "non-monotone," "monotone," or an η with $\mathrm{supp}(\eta) \subseteq B_t$ extending ρ'.

1. Let $C_M > 0$ be a large enough constant; let τ', δ and τ^* be the following parameters:

$$\tau' = (\tau\epsilon/C_M)^2 \cdot \sqrt{\lambda}, \quad \delta = \tau'^2/(C_M \log n) \quad \text{and} \quad \tau^* = \tau'/\sqrt{\lambda}.$$

2. Call `Find-Hi-Influence-Vars`$(f, \rho', \tau', \delta)$ and let H be the set that it returns. If $|H| > 4/\tau'^2$, halt and return "monotone."
3. For each $i \in H$, call `Check-Weight-Positive`$(f, \rho', i, \tau'/2, \delta)$. If any call returns "negative" then halt and output "non-monotone;" if any call returns "fail" then halt and output "monotone;" otherwise (every call returns "positive") continue to Step 4.
4. Repeat $C_M \log n/\sqrt{\lambda}$ times:

   Draw a restriction η with support H, by selecting a random assignment from $\{-1,1\}^H$. Call `Check-Fourier-Regular`$(f_{\rho'\eta}, [n] \setminus \mathrm{supp}(\rho'\eta), \sqrt{C_M\tau^*}, \delta/2)$ and `Estimate-Mean`$(f_{\rho'\eta}, \epsilon/6, \delta/2)$.

   Halt and output the first restriction η where `Check-Fourier-Regular` outputs "regular" and `Estimate-Mean` returns a number of absolute value $\leq 1 - 7\epsilon/6$. If the procedure fails to find such a restriction η, halt and output "monotone."

Figure 5 Subroutine `Maintain-Regular-and-Balanced`. Lemma 24 assumes that $f_{\rho^{(t)}}$ is an $(\tau, \epsilon, \lambda(1 - t/(8\log n)))$-non-monotone LTF, $|A_t| \geq m/4$ and (3), and $f_{\rho'}$ is 0.96-balanced.

▶ **Lemma 23.** *Assume that $f_{\rho^{(t)}}$ is a $(\tau, \epsilon, \lambda(1 - t/(8\log n)))$-non-monotone LTF, and sets A_t and B_t satisfy $|A_t| \geq m/4$ and (3). With probability at least $1/(100\log n)$, the subroutine `Find-Balanced-Restriction` outputs a restriction ρ' with $\mathrm{supp}(\rho') = \mathrm{supp}(\rho^{(t)}) \cup A_t$ such that ρ' extends $\rho^{(t)}$ and $f_{\rho'}$ is 0.96-balanced.*

For Step 2(c) of `Main-Procedure`, the subroutine `Maintain-Regular-and-Balanced` is given in Figure 5. It is very similar to `Regularize-and-Balance` except the number of rounds in Step 4 and the choice of parameters τ' and δ. We leave the proof of the following lemma to the full version. Lemma 21 follows directly from Lemmas 22, 23, and 24.

▶ **Lemma 24.** *Suppose that $f_{\rho^{(t)}}$ is a $(\tau, \epsilon, \lambda(1 - t/(8\log n)))$-non-monotone LTF, sets A_t and B_t satisfy $|A_t| \geq m/4$ and (3), and $f_{\rho'}$ is 0.96-balanced. Then with probability at least $1 - 1/(100\log n)$, `Maintain-Regular-and-Balance` returns either "non-monotone," or a restriction η with $\mathrm{supp}(\eta) \subseteq B_t$ such that $f_{\rho^{(t+1)}}$, where $\rho^{(t+1)} = \rho'\eta$, is $(\tau, \epsilon, \lambda(1 - (t+1)/(8\log n)))$-non-monotone.*

5.4 Final analysis of the algorithm

We conclude by stating the correctness and query complexity of the algorithm. The proofs of the following two theorems appear in Appendix C.

▶ **Theorem 25.** *The algorithm `Mono-Test-LTF`(f, ϵ) correctly tests whether a given LTF is monotone or ϵ-far from monotone.*

▶ **Theorem 26.** *The algorithm `Mono-Test-LTF`(f, ϵ) makes $\tilde{O}(\log^{42} n/\epsilon^{90})$ queries.*

Theorem 1 follows as an immediate consequence of Theorems 25 and 26.

References

1. N. Ailon, B. Chazelle, S. Comandur, and D. Liu. Estimating the distance to a monotone function. *Random Structures and Algorithms*, 31(3):371–383, 2007.
2. Roksana Baleshzar, Deeparnab Chakrabarty, Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and C. Seshadhri. Optimal unateness testers for real-values functions: Adaptivity helps. In *Proceedings of the 44th International Colloquium on Automata, Languages and Programming (ICALP '2017)*, 2017.
3. T. Batu, R. Kumar, and R. Rubinfeld. Sublinear algorithms for testing monotone and unimodal distributions. In *Proceedings of the 36th ACM Symposium on Theory of Computing*, pages 381–390, 2004.
4. A. Belovs and E. Blais. A polynomial lower bound for testing monotonicity. In *Proceedings of the 48th ACM Symposium on Theory of Computing*, 2016.
5. Piotr Berman, Sofya Raskhodnikova, and Grigory Yaroslavtsev. L_p-testing. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*, pages 164–173, 2014.
6. E. Blais, J. Brody, and K. Matulef. Property testing lower bounds via communication complexity. *Computational Complexity*, 21(2):311–358, 2012.
7. E. Blais, S. Raskhodnikova, and G. Yaroslavtsev. Lower bounds for testing properties of functions on hypergrid domains. *Electronic Colloquium on Computational Complexity (ECCC)*, 20:36, 2013.
8. J. Briët, S. Chakraborty, D. García-Soriano, and A. Matsliah. Monotonicity testing and shortest-path routing on the cube. *Combinatorica*, 32(1):35–53, 2012.
9. C. Canonne, D. Ron, and R. Servedio. Testing probability distributions using conditional samples. *SIAM Journal on Comput.*, 44(3):540–616, 2015.
10. D. Chakrabarty and C. Seshadhri. A $o(n)$ monotonicity tester for boolean functions over the hypercube. In *Proceedings of the 45th ACM Symposium on Theory of Computing*, pages 411–418, 2013.
11. D. Chakrabarty and C. Seshadhri. Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids. In *Proceedings of the 45th ACM Symposium on Theory of Computing*, pages 419–428, 2013.
12. D. Chakrabarty and C. Seshadhri. An optimal lower bound for monotonicity testing over hypergrids. *Theory of Computing*, 10(17):453–464, 2014.
13. X. Chen, A. De, R. A. Servedio, and L.-Y. Tan. Boolean function monotonicity testing requires (almost) $n^{1/2}$ non-adaptive queries. In *Proceedings of the 47th ACM Symposium on Theory of Computing*, pages 519–528, 2015.
14. X. Chen, R. A. Servedio, and L.-Y. Tan. New algorithms and lower bounds for monotonicity testing. In *Proceedings of the IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 286–295, 2014.
15. Xi Chen, Rocco A. Servedio, Li-Yang Tan, Erik Waingarten, and Jinyu Xie. Settling the query complexity of non-adaptive junta testing. In *Proceedings of the 32nd Conference on Computational Complexity (CCC '2017)*, 2017.
16. Xi Chen, Erik Waingarten, and Jinyu Xie. Beyond talagrand functions: new lower bounds for testing monotonicity and unateness. In *Proceedings of the 49th ACM Symposium on the Theory of Computing (STOC '2017)*, 2017.
17. Y. Dodis, O. Goldreich, E. Lehman, S. Raskhodnikova, D. Ron, and A. Samorodnitsky. Improved testing algorithms for monotonocity. In *Proceedings of the 3rd International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 97–108, 1999.
18. D. Dzindzalieta. Tight Bernoulli tail probability bounds. Technical Report Doctoral Dissertation, Physical Sciences, Mathematics (01 P), Vilnius University, 2014.

19 F. Ergün, S. Kannan, S. R. Kumar, R. Rubinfeld, and M. Vishwanthan. Spot-checkers. *Journal of Computer and System Sciences*, 60:717–751, 2000.

20 W. Feller. *An introduction to probability theory and its applications*. John Wiley & Sons, 1968.

21 E. Fischer. On the strength of comparisons in property testing. *Information and Computation*, 189(1):107–116, 2004.

22 E. Fischer, E. Lehman, I. Newman, S. Raskhodnikova, R. Rubinfeld, and A. Samorodnitsky. Monotonicity testing over general poset domains. In *Proceedings of the 34th Annual ACM Symposium on the Theory of Computing*, pages 474–483, 2002.

23 O. Goldreich, S. Goldwasser, E. Lehman, D. Ron, and A. Samordinsky. Testing monotonicity. *Combinatorica*, 20(3):301–337, 2000.

24 S. Halevy and E. Kushilevitz. Testing monotonicity over graph products. *Random Structures and Algorithms*, 33(1):44–67, 2008.

25 S. Khot, D. Minzer, and M. Safra. On monotonicity testing and boolean isoperimetric type theorems. In *Proceedings of the 56th Annual Symposium on Foundations of Computer Science*, pages 52–58, 2015.

26 K. Matulef, R. O'Donnell, R. Rubinfeld, and R. Servedio. Testing halfspaces. *SIAM Journal on Comput.*, 39(5):2004–2047, 2010.

27 D. Ron, R. Rubinfeld, M. Safra, A. Samorodnitsky, and O. Weinstein. Approximating the influence of monotone Boolean functions in $O(\sqrt{n})$ query complexity. *ACM Transactions on Computation Theory*, 4(4):1–12, 2012.

28 D. Ron and R. A. Servedio. Exponentially improved algorithms and lower bounds for testing signed majorities. *Algorithmica*, 72(2):400–429, 2015.

29 R. Rubinfeld and R. A. Servedio. Testing monotone high-dimensional distributions. *Random Structures and Algorithms*, 34(1):24–44, 2009. doi:10.1002/rsa.20247.

30 R. A. Servedio, L.-Y. Tan, and J. Wright. Adaptivity helps for testing juntas. In *Proceedings of the 30th IEEE Conference on Computational Complexity*, pages 264–279, 2015.

A Proof of Lemma 19

Proof. Using Lemma 15, with probability $1-\delta$, `Find-Hi-Influence-Vars` in Step 2 returns a set $H \subseteq [n]$ of indices that satisfies the following property:

$$\text{If } |\hat{f}(i)| \geq \tau' \text{ then } i \in H; \text{ If } |\hat{f}(i)| < \tau'/2 \text{ then } i \notin H. \tag{4}$$

When this happens, we have by Parseval $|H| \leq 4/\tau'^2$, and the procedure continues to Step 3.

We consider two subevents: E_0': H satisfies (4) but contains an elements i with $w_i < 0$; and E_0: H satisfies (4) and every $i \in H$ has $w_i > 0$. We have $\Pr[E_0'] + \Pr[E_0] \geq 1-\delta$ as discussed above. Below we show that the procedure returns "non-monotone" with high probability, conditioning on E_0', and it returns a restriction with the desired property with high probability, conditioning on E_0. By the end we combine the two cases to conclude that

$$\begin{aligned}&\Pr[E_0'] \cdot \Pr\left[\text{the procedure returns "non-monotone"} \mid E_0'\right] \\ &\quad + \Pr[E_0] \cdot \Pr\left[\text{it returns } \rho \text{ such that } f_\rho \text{ is } (\tau,\epsilon,\lambda)\text{-non-monotone} \mid E_0\right] \geq 9/10.\end{aligned}$$

We first address the (easier) case of E_0'. Assume $i \in H$ satisfies $w_i < 0$. From (4), $|\hat{f}(i)| \geq \tau'/2$ and thus, `Check-Weight-Positive`$(f, (*)^n, i, \tau'/2, \delta)$ in Step 3 returns "negative" with probability $1-\delta$, and the procedure returns "non-monotone" with probability $1-\delta$, conditioning on E_0'.

Next we address the (harder) case of E_0. First we use E_1 to denote the event that every call to `Check-Weight-Positive` in Step 3 returns the correct answer, i.e., it returns "positive" for every $i \in H$. By a union bound we have $\Pr[E_1 \mid E_0] \geq 1 - 4\delta/\tau'^2$.

Assuming that E_1 happens, the procedure proceeds to Step 4 and we use E_2 to denote the event that `Check-Fourier-Regular` and `Estimate-Mean` return the correct answer, i.e.:

1. `Check-Fourier-Regular` outputs "not regular" if $|\hat{f}_\rho(i)| \geq \sqrt{12\tau'/\epsilon}$ for some $i \in [n] \setminus H$, and outputs "regular" if $|\hat{f}_\rho(i)| \leq 3\tau'/\epsilon$ for all $i \in [n] \setminus H$, for every ρ in Step 4, and
2. `Estimate-Mean` returns a number a with $|a - \mathbf{E}[f_\rho]| \leq \epsilon/6$, for every ρ in Step 4.

We also write E_3 to denote the event that one of the restrictions ρ drawn in Step 4 satisfies that f_ρ is both $(2\epsilon/3)$-far from monotone and $(3\tau'/\epsilon)$-Fourier-regular. By a union bound, we have that $\Pr[E_2 \mid E_0 \wedge E_1] \geq 1 - C_{RB}\delta/\epsilon$.

In the rest of the proof we show that

1. $\Pr[E_3 \mid E_0 \wedge E_1] \geq 99/100$ and
2. Given E_0, E_1, E_2 and E_3, the procedure always returns a restriction ρ such that f_ρ is $(\tau, \epsilon, \lambda)$-non-monotone.

Together we have that it returns such a ρ with probability at least (conditioning on E_0)

$$(1 - 4\delta/\tau'^2) \cdot (1 - C_{RB}\delta/\epsilon - 1/100).$$

Summarizing the two cases of E_0' and E_0 we have that `Regularize-and-Balance` returns either "non-monotone" or a ρ such that f_ρ is $(\tau, \epsilon, \lambda)$-non-monotone with probability at least

$$\Pr[E_0'] \cdot (1 - \delta) + \Pr[E_0] \cdot (1 - 4\delta/\tau'^2) \cdot (1 - C_{RB}\delta/\epsilon - 1/100) > 9/10,$$

using $\Pr[E_0'] + \Pr[E_0] \geq 1 - \delta$ and our choice of δ (by letting C_{RB} be large enough).

We use the following claim to show that $\Pr[E_3 \mid E_0 \wedge E_1] \geq 99/100$.

▶ **Claim 27.** *A random restriction ρ over H satisfies that f_ρ is both $(2\epsilon/3)$-far from monotone and $(3\tau'/\epsilon)$-Fourier-regular with probability at least $\epsilon/3$.*

Proof. For each of the two properties, we have

1. Proposition 11: with probability at least $1 - (\epsilon/3)$, f_ρ is $(3\tau'/\epsilon)$-Fourier-regular.
2. Lemma 14: with probability at least $2\epsilon/3$, f_ρ is $(2\epsilon/3)$-far from monotone. To see this, let c be the probability of f_ρ being $(2\epsilon/3)$-far from monotone. Then $c \geq 2\epsilon/3$ follows from

$$(1 - c) \cdot (2\epsilon/3) + c \cdot (1/2) \geq \epsilon,$$

where we used the fact that distance to monotonicity is always at most $1/2$.

The claim then follows from a union bound. ◀

By choosing C_{RB} to be a large enough constant, we have $\Pr[E_3 \mid E_0 \wedge E_1] \geq 99/100$.

Finally we show that conditioning on all four events E_0, E_1, E_2, E_3 the procedure always returns a restriction ρ such that f_ρ is a $(\tau, \epsilon, \lambda)$-non-monotone LTF. We do this in two steps:

1. First, given E_3, one of the restrictions ρ drawn in Step 4 is both $(2\epsilon/3)$-far from monotone and $(3\tau'/\epsilon)$-Fourier-regular. Given E_2, ρ must pass both tests, i.e., `Check-Fourier-Regular` outputs "regular" and `Estimate-Mean` returns a number of absolute value at most $1 - 7\epsilon/6$ in Step 4. The former is trivial; to see the latter, note that being $(2\epsilon/3)$-far from monotone implies that $|\mathbf{E}[f_\rho]| \leq 1 - 4\epsilon/3$ and therefore, the number returned by `Estimate-Mean` is at most $1 - 7\epsilon/6$, given E_2.

2. Second, we show that if a restriction ρ passes both tests in Step 4 of the procedure, then f_ρ must be $(\tau, \epsilon, \lambda)$-non-monotone. One can think of this as a soundness property, saying that if the procedure halts and returns some ρ, that it returns a correct one. To see this, note that by E_2, f_ρ is both $\sqrt{12\tau'/\epsilon}$-Fourier regular and ϵ-balanced. By Theorem 10, f_ρ is $O(\sqrt{\tau'/\epsilon^3})$-weight-regular, and τ-weight-regular by letting C_{RB} be large enough. It also follows from Lemma 12 that f_ρ has λ-significant squared negative weights.

This finishes the proof of the lemma. ◀

B Proofs of Lemma 20, Lemma 22, and Lemma 23

B.1 Proof of the Second Part of Lemma 20 using Lemma 21

Proof. We consider the event E where the conclusion of Lemma 21 holds for every iteration of the while loop of `Main-Procedure`. As the condition of Lemma 21 holds for the first loop ($t = 0$) and there are at most $4 \log n$ many loops, this happens with probability at least $9/10$. Since E implies $|A_t| \geq |\textsc{stars}(\rho^{(t)})|/4$, we can also assume that the procedure never halts and outputs "monotone" due to line 2(d).

Given E, `Main-Procedure` either returns "non-monotone" as desired or reaches line 3. Furthermore, if it reaches line 3, $f_{\rho^{(t)}}$ must be $(\tau, \epsilon, \lambda/2)$-non-monotone by Lemma 21 and have at most $1/\tau^2$ variables. It follows from Lemma 13 that $f_{\rho^{(t)}}$ is ϵ'-far from monotone, where $\epsilon' = \epsilon^3/(C \log(1/\epsilon))$ for some large enough constant C. Finally, by Theorem 6, `Edge-Tester` outputs "non-monotone" (by finding an anti-monotone edge) with probability at least $9/10$ and the proof is complete. ◀

B.2 Proof of Lemma 22

Proof. We consider the three events separately and then apply a union bound.

First by Chernoff bound, $|A_t| \geq m/4$ holds with probability at least $1 - e^{-\Omega(m)}$.

Next for the first inequality in (3), assume without loss of generality that $\sum_{i \in I} w_i^2 = 1$ (as $f_{\rho^{(t)}}$ cannot be all-1 or all-(-1)). By Hoeffding bound the probability that it does not hold is at most

$$2 \exp\left(-\Omega\left(\frac{1/\log^2 n}{\sum_{i \in I} w_i^4}\right)\right).$$

Since $f_{\rho^{(t)}}$ is τ-weight-regular (over I), we have that $|w_i| \leq \tau$ for all $i \in I$ and thus,

$$\sum_{i \in I} w_i^4 \leq \tau^2 \cdot \sum_{i \in I} w_i^2 = \tau^2.$$

As a result, the second inequality holds with probability at least $1 - \exp(-\Omega(1/(\tau^2 \log^2 n)))$.

For the last inequality, note that $\sum_{i \in I : w_i < 0} w_i^2 \geq \lambda(1 - t/(8 \log n))$. Similarly by Hoeffding,

$$\Pr\left[\sum_{i \in B_t : w_i < 0} w_i^2 < \left(\frac{\lambda}{2}\right)\left(1 - \frac{t + 0.5}{8 \log n}\right)\right] \leq \exp\left(-\Omega\left(\frac{\lambda^2/\log^2 n}{\sum_{i \in I : w_i < 0} w_i^4}\right)\right) \leq \exp\left(-\Omega\left(\log^2 n\right)\right).$$

Combining the above with the analysis of the first inequality in (3), the last inequality holds with probability at least $1 - \exp(-\Omega(\log^2 n))$. The lemma follows from a union bound. ◀

B.3 Proof of Lemma 23

Proof. For convenience we use f' to denote $f_{\rho^{(t)}}$, w' to denote the weight vector w but restricted on I, and θ' to denote the new threshold, i.e.,

$$\theta' = \theta - \sum_{i \in \mathrm{supp}(\rho^{(t)})} \rho^{(t)}(i) \cdot w_i.$$

Without loss of generality we assume that $\sum_{i \in I} w_i'^2 = 1$. We may additionally assume that $\theta' \geq 0$. This assumption is without loss of generality, because 1) if ρ' is a 0.96-balanced restriction when $-\theta' \geq 0$, then $-\rho'$ is a 0.96-balanced restriction for $\theta' \leq 0$, and 2) `Find-Balanced-Restriction` will test the only take into account the absolute value of the output of `Estimate-Mean`. Let

$$\alpha = \sum_{i \in A_t} w_i'^2 \quad \text{and} \quad \beta = \sum_{i \in B_t} w_i'^2.$$

We use $a = b \pm c$ to denote the inequalities $b - c \leq a \leq b + c$. Then from (3) we have that $\alpha, \beta = 1/2 \pm O(1/\log n)$. By assumption, f' is τ-weight-regular and ϵ-balanced.

For the analysis we define two events E_1 and E_2. Here E_1 denotes the event that every call to `Estimate-Mean` returns a number a such that $|a - \mathbf{E}[f_{\rho'}]| \leq 0.01$. By a union bound, this happens with probability $1 - 1/(200 \log n)$. Let E_2 be the event that one of the restrictions ρ^* drawn has f'_{ρ^*} being 0.98-balanced. When E_1 and E_2 both occur, the subroutine outputs a restriction ρ' such that $f_{\rho'}$ is 0.96-balanced. In the rest of the proof we show that event E_2 happens with high probability.

To analyze the probability of f'_{ρ^*} being 0.98-balanced, we use $\mathbf{x}_i$ to denote an independent and unbiased random $\{-1, 1\}$-variable for each $i \in I$, and let

$$\mathbf{x}_A = \sum_{i \in A_t} \mathbf{x}_i \cdot w'_i, \quad \mathbf{x}_B = \sum_{i \in B_t} \mathbf{x}_i \cdot w'_i \quad \text{and} \quad \mathbf{x} = \mathbf{x}_A + \mathbf{x}_B.$$

By Hoeffding bound and the assumption that f' is ϵ-balanced, we have

$$2\epsilon = \Pr[\mathbf{x} \geq \theta'] \leq \exp(-\theta'^2/2). \tag{5}$$

Using Berry–Esséen $\mathbf{x}_A + \mathbf{x}_B$ is $O(\tau)$-close to a standard $\mathcal{N}(0,1)$ Gaussian random variable, denoted by $\mathcal{G}$, $\mathbf{x}_A$ is $O(\tau)$-close to $\sqrt{\alpha}\mathcal{G}$, and $\mathbf{x}_B$ is $O(\tau)$-close to $\sqrt{\beta}\mathcal{G}$.

Let $\theta^* > 0$ be the threshold such that $\Pr[|\sqrt{\beta}\mathcal{G}| \leq \theta^*] = 0.01$. Then

$$\Pr\left[f'_{\rho^*} \text{ is 0.98-balanced}\right] \geq \Pr\left[\mathbf{x}_A \in [\theta' - \theta^*, \theta' + \theta^*]\right].$$

This is because, for any number $x_A \in [\theta' - \theta^*, \theta' + \theta^*]$, we have

$$0.495 - O(\tau) \leq \Pr\left[\mathbf{x}_B \geq \theta' - x_A\right] = \Pr\left[\sqrt{\beta}\mathcal{G} \geq \theta' - x_A\right] \pm O(\tau) \leq 0.505 + O(\tau),$$

in which case the function f'_{ρ^*} is $0.99 - O(\tau) = 0.98$-balanced. To bound $\Pr\left[\mathbf{x}_A \in [\theta' - \theta^*, \theta' + \theta^*]\right]$, we note that $\theta' \geq 0$ (by assumption) and $\theta^* = \Omega(1)$ (by our choice of θ^* and $\beta > 1/3$). As a result,

$$\Pr\left[\mathbf{x}_A \in [\theta' - \theta^*, \theta']\right] \geq \Pr\left[\sqrt{\alpha}\mathcal{G} \in [\theta' - \theta^*, \theta']\right] - O(\tau) = \Omega(1) \cdot \Omega(\epsilon^3) - O(\tau) = \Omega(\epsilon^3),$$

where we used $\alpha > 1/3$ by (3), $\tau = o(\epsilon^3)$, and $\exp(-\theta'^2/2) = \Omega(\epsilon)$ from (5) to obtain

$$\min\left(\exp\left(-(\theta^*)^2/(2\alpha)\right), \exp\left(-\theta'^2/(2\alpha)\right)\right) = \Omega(\epsilon^3).$$

As a result, a random restriction ρ^* is 0.98-balanced with probability at least $\Omega(\epsilon^3)$. Thus with probability $1-1/n$ (by choosing a large enough constant C_{BR}), `Find-Balanced-Restriction` gets such a restriction that would pass the `Estimate-Mean` test. By a union bound on E_1 and E_2, `Find-Balanced-Restriction` returns a 0.96-balanced ρ' with probability at least $1 - 1/(200 \log n) - 1/n > 1 - 1/(100 \log n)$. This finishes the proof of the lemma. ◀

C Proofs of the Final Analysis

Proof of Theorem 25. The algorithm is one-sided because it outputs "non-monotone" only when an anti-monotone edge is found. The only interesting case is when the input LTF f is ϵ-far from monotone. Combining Lemmas 19 and 20, the algorithm `Mono-Test-LTF`(f, ϵ) outputs "non-monotone" with probability at least $(9/10)(81/100) > 2/3$. This completes the proof. ◀

Proof of Theorem 26. From Fact 18, the number of queries used by `Regularize-and-Balance` is $\tilde{O}(\log^{41} n/\epsilon^{90})$, since the main bottleneck is the call to `Find-Hi-Influence-Vars`. In `Main-Procedure`, the bottleneck is the $O(\log n)$ calls to `Find-Hi-Influence-Vars` in `Maintain-Regular-and-Balance`, each of query complexity $\tilde{O}(\log^{41} n/\epsilon^{90})$, despite the slightly different parameters. Note that we run the edge tester when there are fewer than $1/\tau^2$ many stars, so it makes $\tilde{O}\left(\log^4 n/\epsilon^9\right)$ many queries. ◀