Testing Low-Degree Polynomials over $GF(2)$

Noga Alon[1 ⋆], Tali Kaufman[2 ⋆⋆], Michael Krivelevich[3 ⋆⋆⋆], Simon Litsyn[4 †], and Dana Ron[4 ‡]

[1] Department of Mathematics, Tel Aviv University, Tel Aviv 69978, Israel and Institute for Advanced Study, Princeton, NJ 08540, USA.
[2] School of Computer Science, Tel Aviv University,Tel Aviv 69978 Israel.
[3] Department of Mathematics, Tel Aviv University, Tel Aviv 69978, Israel.
[4] Department of Electrical Engineering-Systems, Tel Aviv University, Tel Aviv 69978, Israel.

Abstract. We describe an efficient randomized algorithm to test if a given *binary* function $f : \{0,1\}^n \to \{0,1\}$ is a low-degree polynomial (that is, a sum of low-degree monomials). For a given integer $k \geq 1$ and a given real $\epsilon > 0$, the algorithm queries f at $O(\frac{1}{\epsilon} + k4^k)$ points. If f is a polynomial of degree at most k, the algorithm always accepts, and if the value of f has to be modified on at least an ϵ fraction of all inputs in order to transform it to such a polynomial, then the algorithm rejects with probability at least $2/3$. Our result is essentially tight: Any algorithm for testing degree-k polynomials over $GF(2)$ must perform $\Omega(\frac{1}{\epsilon} + 2^k)$ queries.

1 Introduction

In this work we consider the problem of testing whether a binary function $f : \{0,1\}^n \to \{0,1\}$ is a polynomial of degree at most k satisfying $f(0,\ldots,0) = 0$, for a given integer parameter k. Such a polynomial is simply a sum (modulo 2) of monomials each being a product of at most k variables, with the free term equal to zero. (The restriction $f(0,\ldots,0) = 0$ is imposed mainly for historical reasons, to make our definition and result consistent with the previously treated case of linear functions $k = 1$. With minor changes our algorithm can be adapted to test the class of all polynomials of degree at most k in n variables, without the restriction on the free term.) The algorithm is required to accept functions that are polynomials of degree at most k (vanishing at zero), and to reject, with probability at least $2/3$, functions that are *far* from any such polynomial. More precisely, the algorithm is given a distance parameter ϵ, and is required

⋆ E-mail: nogaa@post.tau.ac.il. Research supported in part by a USA Israeli BSF grant and by a grant from the Israel Science Foundation

⋆⋆ E-mail: kaufmant@post.tau.ac.il, This work is part of the author's Ph.D. thesis prepared at Tel Aviv University under the supervision of Prof. Noga Alon, and Prof. Michael Krivelevich.

⋆⋆⋆ E-mail: krivelev@post.tau.ac.il. Research supported in part by a USA Israeli BSF grant and by a grant from the Israel Science Foundation.

† E-mail: litsyn@eng.tau.ac.il. Research supported in part by a USA Israeli BSF grant and by a grant from the Israel Science Foundation.

‡ E-mail: danar@eng.tau.ac.il. Research supported by the Israel Science Foundation (grant number 32/00-1).

to reject (with probability at least $2/3$) any function whose value should be modified on more that an ϵ-fraction of the domain to become a degree-k polynomial f satisfying $f(0,\ldots,0) = 0$. To this end the algorithm can query the function f on inputs of its choice, where our goal is to minimize the query complexity of the algorithm (as a function of k, $1/\epsilon$, and n).

The problem of testing multivariate low-degree polynomials has been studied quite extensively [4, 3, 13, 11, 17, 12, 2], and has important applications in the context of Probabilistically Checkable Proofs (PCP). However, with the exception of the case $k = 1$, that is, linear functions (which we discuss below), all results apply only to testing polynomials over fields that *are larger than* k (the degree bound). When the field F is sufficiently large, it is possible to reduce the problem of testing whether a function $f : F^n \to F$ is a multivariate degree-k polynomial to testing whether a function is a degree-k *univariate* polynomial, where the latter task is simply based on interpolation. Namely, the test for f selects random *lines* in F^n (more precisely, in the finite projective geometry PG$(n-1, |F|)$), and verifies that the restriction of f to each of these lines is a (univariate) polynomial of degree at most k. This reduction does not hold for small fields, and in particular for $GF(2)$, which is our focus.

As noted above, in the case of $k = 1$ (linear functions), the linearity test of Blum, Luby and Rubinfeld [10] works also when the underlying field is $GF(2)$. In fact, our test can be viewed as an extension of the [10] algorithm, as we explain in more detail below. Linearity testing has also been studied in the following papers [4, 11, 6, 7, 5].

Our Results

We describe and analyze an algorithm that tests whether a function $f : \{0,1\}^n \to \{0,1\}$ is a degree-k polynomial satisfying $f(0,\ldots,0) = 0$, or is ϵ-far from any such polynomial, using $O(1/\epsilon + k \cdot 2^{2k})$ queries. As we show, the exponential dependency on k is unavoidable. This is in contrast to the case of testing degree-k polynomials over larger fields, where the sample complexity is polynomial in k. Our testing algorithm is simple. It repeats the following check $\Theta(\frac{1}{2^k \epsilon} + k2^k)$ times: It selects, uniformly and at random, $k+1$ vectors $y_1, \ldots, y_{k+1} \in \{0,1\}^n$. It then evaluates f on the sum of every non-empty subset of the selected vectors, and checks that the sum of these evaluations is 0. If all checks succeed then it accepts, otherwise it rejects. Note that for the special case of $k = 1$, we obtain the linearity test of [10] which uniformly selects $O(1/\epsilon)$ pairs $y_1, y_2 \in \{0,1\}^n$, and verifies for each pair that $f(y_1) + f(y_2) = f(y_1 + y_2)$.

Our choice of the sets corresponds to a random selection of a $(k+1)$-dimensional subspace in the affine geometry AG$(n, 2)$ (see for example [14, Chap. 12]). In case $k = 1$ we deal with lines of the affine geometry PG$(n, 2)$.

As a by-product of our analysis we obtain a *self-corrector* (as defined in [10]) for f, in case f is sufficiently close to a degree-k polynomial g. Specifically, for any given $x \in \{0,1\}^n$, it is possible to obtain the value $g(x)$ with high probability by querying f on additional, randomly selected, points.

Relation to Coding

Our setting and results have a very natural interpretation in terms of coding theory. The set of (evaluations of) all polynomials in n variables of degree at most k over $GF(2)$ is called the *Reed-Muller code* $\mathcal{R}(k,n)$ with parameters k and n. (See, e.g., [16] for relevant background). So our algorithm can be considered as (locally) testing Reed-Muller codes. To be more accurate, as we consider only polynomials f vanishing at zero, we in fact test the so-called *shortened Reed-Muller code* $\mathcal{R}(k,n)^*$, obtained from $\mathcal{R}(k,n)$ by choosing all codewords with the first bit (i.e. that corresponding to the zero vector) equal to zero, and deleting this bit. The Reed-Muller code $\mathcal{R}(k,n)$ is a linear code in $\{0,1\}^{2^n}$ of minimum distance 2^{n-k}. The dual code of $\mathcal{R}(k,n)$ is the Reed-Muller code $\mathcal{R}(n-k-1,n)$. The dual code of the shortened Reed-Muller code $\mathcal{R}(k,n)^*$ is the so called *punctured* Reed-Muller code with parameters $n-k-1$ and n, obtained from $\mathcal{R}(n-k-1,n)$ by deleting the first bit of every codeword. The minimum distance of the punctured Reed-Muller code with parameters $n-k-1$ and n is $2^{k+1}-1$, and its minimum weight codewords are obtained from the minimum weight codewords of $\mathcal{R}(n-k-1,n)$, having the first bit equal to 1, by deleting this bit; the number of minimum weight vectors is proportional to $2^{(k+1)n}$.

For an arbitrary vector from $\{0,1\}^{2^n}$ we want to distinguish between two cases: the vector belongs to the code, or, alternatively, it is at (Hamming) distance at least $\epsilon \cdot 2^n$ from the closest codeword of $\mathcal{R}(k,n)^*$. Our strategy is then to pick a random minimum weight vector from the punctured $\mathcal{R}(n-k-1,n)$, and to check if it is orthogonal to the tested vector. Clearly, this will always confirm orthogonality if the considered vector is from the code. However, we prove that if the tested vector is far enough from the code, with positive probability the test will detect it, and give an estimate for this probability.

2 Preliminaries

For any integer ℓ, we denote by $[\ell]$ the set $\{1,\ldots,\ell\}$. For any $k \in [n]$, let $\mathcal{P}_k$ denote the family of all Boolean functions over $\{0,1\}^n$ which are polynomials of degree at most k without a free term. That is, $f \in \mathcal{P}_k$ if and only if there exist coefficients $a_S \in \{0,1\}$, for every $S \subseteq [n], 1 \le |S| \le k$, such that

$$f = \sum_{S\subseteq[n],|S|\in[k]} a_S \cdot \prod_{i\in S} x_i \ , \tag{1}$$

where the addition is in $GF(2)$. In particular, $\mathcal{P}_1$ is the family of all linear functions over $\{0,1\}^n$, that is, all functions of the form $\sum_{i\in S} x_i$, where $S \subseteq [n]$.

For any two functions $f, g : \{0,1\}^n \to \{0,1\}$, the symmetric difference between f and g is $\Delta(f,g) \stackrel{\text{def}}{=} \{y \in \{0,1\}^n : f(y) \neq g(y)\}$. The relative distance $\mathrm{dist}(f,g) \in [0,1]$ between f and g is: $\mathrm{dist}(f,g) \stackrel{\text{def}}{=} |\Delta(f,g)|/2^n$. For a function g and a family of functions F, we say that g is ϵ-far from F, for some $0 < \epsilon < 1$, if, for every $f \in F$, $\mathrm{dist}(g,f) > \epsilon$. Otherwise it is ϵ-close to F.

A testing algorithm (tester) for $\mathcal{P}_k$ is a probabilistic algorithm, that is given query access to a function f, and a distance parameter ϵ, $0 < \epsilon < 1$. If f belongs to $\mathcal{P}_k$ then with probability at least $\frac{2}{3}$, the tester should accept f, and if f is ϵ-far from $\mathcal{P}_k$, then with probability at least $\frac{2}{3}$ the tester should reject it. If the tester accepts every f in $\mathcal{P}_k$ with probability 1, then it is a one-sided tester.

The following notation will be used extensively in this paper. Given a function $f : \{0,1\}^n \to \{0,1\}$, for $y_1, ..., y_\ell \in \{0,1\}^n$ let

$$T_f(y_1, \ldots, y_\ell) \stackrel{\text{def}}{=} \sum_{\emptyset \neq S \subseteq [\ell]} f\left(\sum_{i \in S} y_i\right) , \tag{2}$$

where the first sum is over $GF(2)$ and the second one is over $(GF(2))^n$, and let

$$T_f^{y_1}(y_2, \ldots, y_\ell) \stackrel{\text{def}}{=} T_f(y_1, \ldots, y_\ell) + f(y_1) . \tag{3}$$

3 Characterization of Low Degree Polynomials over $\{0,1\}^n$

Claim 1 *A function f belongs to $\mathcal{P}_k$ (i.e., it is a polynomial of total degree at most k satisfying $f(0,0,\ldots,0) = 0$), if and only if* for every $y_1, \ldots, y_{k+1} \in \{0,1\}^n$ *we have*

$$T_f(y_1, \ldots, y_{k+1}) = 0 . \tag{4}$$

Proof. A polynomial from $\mathcal{P}_k$ can be viewed as a code word in the appropriate Reed-Muller code, see, e.g., [16]. Thus, the above characterization can be proved using known facts about its dual. For completeness we provide a direct, simple proof.

We first prove that if a function f belongs to $\mathcal{P}_k$ then $T_f(y_1, \ldots, y_{k+1}) = 0$ for every $y_1, \ldots, y_{k+1} \in \{0,1\}^n$.
As f is a sum of monomials of total degree at most k it suffices to show that for every monomial $m = \prod_{i \in I} x_i$, where $1 \leq |I| \leq k$, $T_m(y_1, \ldots, y_{k+1}) = 0$ for every $y_1, \ldots, y_{k+1} \in \{0,1\}^n$. The number of linear combinations $\sum_{j=1}^{k+1} b_j y_j$, where $b_j \in \{0,1\}$, for which $m(\sum_{j=1}^{k+1} b_j y_j) = 1$ is clearly the number of solutions of a linear system of $|I|$ equations in the $k+1$ variables b_j, and the trivial combination $b_j = 0$ for all j is not one of the solutions. Therefore, this number of solutions (which is possibly zero) is divisible by $2^{k+1-|I|}$, showing that there is an even number of sets S satisfying $\emptyset \neq S \subset [k+1]$ such that $m(\sum_{i \in S} y_i) = 1$. This implies that $T_m(y_1, \ldots, y_{k+1}) = 0$, as needed.
We next show that if $f = f(x_1, x_2, \ldots, x_n) : \{0,1\}^n \mapsto \{0,1\}$ satisfies Equation (4) for every $y_1, y_2, \ldots, y_{k+1} \in \{0,1\}^n$, then $f \in \mathcal{P}_k$. Every function from $\{0,1\}^n$ to $\{0,1\}$ can be written uniquely as a polynomial over $GF(2)$:

$$f = \sum_{I \subset [n]} a_I \prod_{i \in I} x_i .$$

Our objective is to show that $a_\emptyset = 0$ and that $a_I = 0$ for all $|I| > k$. Taking $y_j = (0, 0, \ldots, 0)$ for every j we conclude, by (4), that $a_\emptyset = 0$. Suppose, now, that there is a nonzero a_I with $|I| > k$. Take such an I of minimum cardinality, and assume, without loss of generality, that $I = [s]$ with $s \geq k+1$.

Let e_i denote the i-th unit vector in $\{0,1\}^n$, and define $y_1 = e_1, y_2 = e_2, \ldots, y_k = e_k$ and $y_{k+1} = e_{k+1} + \ldots + e_s$. Then the monomial $m = a_I \prod_{i \in I} x_i$ does not vanish on $\sum_{i=1}^{k+1} y_i$ and does vanish on $\sum_{i \in S} y_i$ for every $\emptyset \neq S \neq [k+1]$. Thus $T_m(y_1, \ldots, y_{k+1}) \neq 0$. On the other hand, for any other monomial, say, $m' = \prod_{i \in I'} x_i$ with a nonzero coefficient in the representation of f, $T_{m'}(y_1, \ldots, y_{k+1}) = 0$. Indeed, if $|I'| \leq k$ this holds by the first part of the proof. Otherwise, by the minimality of I, $m'(\sum_{i \in S} y_i) = 0$ for all $S \subset [k+1]$. Altogether this implies that $T_f(y_1, y_2, \ldots, y_{k+1}) = 1$, contradicting the assumption.
This completes the proof of Claim 1.

4 A One-Sided Tester for Low Degree Polynomials over $\{0,1\}^n$

In this section we present and analyze a one-sided tester for $\mathcal{P}_k$. This tester generalizes the linearity tester of Blum, Luby and Rubinfeld [10].

Algorithm Test-$\mathcal{P}_k$

1. Uniformly and independently select $\Theta(\frac{1}{2^k \epsilon} + k2^k)$ groups of vectors. Each group contains $k+1$ uniformly selected random vectors $y_1, \ldots, y_{k+1} \in \{0,1\}^n$.
2. If for some group of vectors $y_1, \ldots, y_{k+1}$ it holds that $T_f(y_1, \ldots, y_{k+1}) \neq 0$, then reject, otherwise, accept.

Theorem 1 *The algorithm* **Test-$\mathcal{P}_k$** *is a one-sided tester for $\mathcal{P}_k$ with query complexity $\Theta(\frac{1}{\epsilon} + k2^{2k})$.*

From the test definition and from Claim 1 it is obvious that if $f \in \mathcal{P}_k$, then the tester accepts. Thus, the crux of the proof is to show that if f is ϵ-far from $\mathcal{P}_k$, then the tester rejects with probability at least $2/3$. Our proof has a similar general structure to Sudan's analysis [18] of the linearity test in [10], but requires some additional ideas. In particular, if f is the function tested, we can define a function g as follows. For any $y \in \{0,1\}^n$:

$$g(y) = 1 \text{ if } \Pr_{y_2,\ldots,y_{k+1} \in \{0,1\}^n}[T_f^y(y_2, \ldots, y_{k+1}) = 1] \geq 1/2 \text{ and } g(y) = 0 \text{ otherwise.} \tag{5}$$

Thus g is a kind of *majority* function. That is, for every vector $y \in \{0,1\}^n$, $g(y)$ is chosen to satisfy most of the equations $T_f^y(y_2, \ldots, y_{k+1}) = g(y)$. We also define

$$\begin{aligned}\eta &\stackrel{\text{def}}{=} \Pr_{y_1,\ldots,y_{k+1} \in \{0,1\}^n}[T_f(y_1, \ldots, y_{k+1}) \neq 0] \\ &= \Pr_{y_1,\ldots,y_{k+1} \in \{0,1\}^n}[T_f^{y_1}(y_2, \ldots, y_{k+1}) \neq f(y_1)] \,. \end{aligned} \tag{6}$$

Note that η is simply the probability that a single group of vectors $y_1, \ldots, y_{k+1}$ selected by the algorithm provides evidence that $f \notin \mathcal{P}_k$. We shall prove two claims. The first, and simpler claim (in Lemma 2), is that if η is small, then g is close to f. The second and more involved claim (in Lemma 5) is that if η is small, then g must belong to $\mathcal{P}_k$. This would suffice for proving the correctness of a slight variation on our algorithm that uses a larger sample size. In order to attain the sample complexity claimed in Theorem 1, we shall need to prove one more claim that deals with the case in which η is very small (see Lemma 6).

Lemma 2 *For a fixed function f, let g and η be as defined in Equations (5) and (6), respectively. Then,* $\mathrm{dist}(f,g) \le 2\eta$.

Proof. Recall that for every $y \in \{0,1\}^n$, $\Pr_{y_2,\ldots,y_{k+1}\in\{0,1\}^n}[T_f^y(y_2,\ldots,y_{k+1}) = g(y)] \ge 1/2$. Hence

$$
\begin{aligned}
\eta &= \Pr_{y,\,y_2,\ldots,y_{k+1}\in\{0,1\}^n}[T_f^y(y_2,\ldots,y_{k+1}) \ne f(y)] \\
&= \frac{1}{2^n}\sum_{y\in\{0,1\}^n} \Pr_{y_2,\ldots,y_{k+1}\in\{0,1\}^n}[T_f^y(y_2,\ldots,y_{k+1}) \ne f(y)] \\
&\ge \frac{1}{2^n}\sum_{y\in\Delta(f,g)} \Pr_{y_2,\ldots,y_{k+1}\in\{0,1\}^n}[T_f^y(y_2,\ldots,y_{k+1}) = g(y)] \\
&\ge \frac{1}{2^n}\cdot|\Delta(f,g)|\cdot\frac{1}{2}
\end{aligned}
$$

Thus, $\mathrm{dist}(f,g) = \frac{|\Delta(f,g)|}{2^n} \le 2\eta$.

Recall that by the definition of g as a majority function, for every y, we have that for at least one half of the k-tuples of vectors $y_2,\ldots,y_{k+1}$, $T_f^y(y_2,\ldots,y_{k+1}) = g(y)$. In the next lemma we show that this equality actually holds for a vast majority of the k-tuples $y_2,\ldots,y_{k+1}$ (assuming η is sufficiently small).

Lemma 3 *For every* $y \in \{0,1\}^n$: $\Pr_{y_2,\ldots,y_{k+1}\in\{0,1\}^n}[g(y) = T_f^y(y_2,\ldots,y_{k+1})] \ge 1 - 2k\eta$.

In order to prove Lemma 3 we shall first establish the following claim.

Claim 4 *For every* $y,z,w,y_2,\ldots,y_k \in \{0,1\}^n$,

$$
\begin{aligned}
&T_f(y,y_2,\ldots,y_k,w) + T_f(y,y_2,\ldots,y_k,z) \\
&\quad = T_f(y+w,y_2,\ldots,y_k,y+w+z) + T_f(y+z,y_2,\ldots,y_k,y+w+z) \qquad (7)
\end{aligned}
$$

Proof. Let $Y = \{y_2,\ldots,y_k\}$, and consider any set $I \subseteq \{2,\ldots,k\}$, which may be the empty set. For a vector $x \in \{0,1\}^n$ denote $f_{Y,I}(x) \stackrel{\mathrm{def}}{=} f(\sum_{i\in I} y_i + x)$.
For every set $I \subseteq \{2,\ldots,k\}$, each element of type $f(\sum_{i\in I} y_i)$ appears twice in both sides of Equation (7) and thus cancels out. Now for every set $I \subset \{2,\ldots,k\}$ (including the empty set), we get in the left hand side of Equation (7):

$$f_{Y,I}(y) + f_{Y,I}(w) + f_{Y,I}(y+w) + f_{Y,I}(y) + f_{Y_I}(z) + f_{Y,I}(y+z)\ .$$

In the right hand side of Equation (7) we get:

$$f_{Y,I}(y+w) + f_{Y,I}(y+z+w) + f_{Y,I}(z) + f_{Y,I}(y+z) + f_{Y,I}(y+w+z) + f_{Y,I}(w)\ .$$

This implies equality over $GF(2)$.

We now turn to prove Lemma 3.

Proof of Lemma 3: We fix $y \in \{0,1\}^n$ and let $\gamma \stackrel{\text{def}}{=} \Pr_{y_2,\ldots,y_{k+1}\in\{0,1\}^n}[g(y) = T_f^y(y_2,\ldots,y_{k+1})]$. Recall that we are interested in proving that $\gamma \geq 1-2k\eta$. To this end, we shall bound a slightly different, but related probability. Let

$$\delta \stackrel{\text{def}}{=} \Pr_{y_2,\ldots,y_{k+1},z_2,\ldots,z_{k+1}\in\{0,1\}^n}[T_f^y(y_2,\ldots,y_{k+1}) = T_f^y(z_2,\ldots,z_{k+1})] \,. \tag{8}$$

Then, by the definitions of γ and δ,

$$\begin{aligned}\delta &= \Pr[T_f^y(y_2,\ldots,y_{k+1}) = g(y) \text{ and } T_f^y(z_2,\ldots,z_{k+1}) = g(y)] \\ &\qquad + \Pr[T_f^y(y_2,\ldots,y_{k+1}) \neq g(y) \text{ and } T_f^y(z_2,\ldots,z_{k+1}) \neq g(y)] \\ &= \gamma^2 + (1-\gamma)^2 \end{aligned} \tag{9}$$

where the probabilitites are over the choice of $y_2,\ldots,y_{k+1},z_2,\ldots,z_{k+1} \in \{0,1\}^n$. Since we are working over $GF(2)$,

$$\delta = \Pr_{y_2,\ldots,y_{k+1},z_2,\ldots,z_{k+1}\in\{0,1\}^n}[T_f(y,y_2,\ldots,y_{k+1}) + T_f(y,z_2,\ldots,z_{k+1}) = 0] \,.$$

Now, for any choice of $y_2,\ldots,y_{k+1}$ and $z_2,\ldots,z_{k+1}$:

$$\begin{array}{lll} T_f(y,y_2,\ldots,y_{k+1}) & + T_f(y,z_2,\ldots,z_{k+1}) & = \\ T_f(y,y_2,\ldots,y_{k+1}) & + T_f(y,y_2,\ldots,y_k,z_{k+1}) & + \\ T_f(y,y_2,\ldots,y_k,z_{k+1}) & + T_f(y,y_2,\ldots,y_{k-1},z_k,z_{k+1}) & + \\ T_f(y,y_2,\ldots,y_{k-1},z_k,z_{k+1}) & + T_f(y,y_2,\ldots,y_{k-2},z_{k-1},z_k,z_{k+1}) & + \\ . & & \\ . & & \\ . & + & \\ T_f(y,y_2,z_3,\ldots,z_{k+1}) & + T_f(y,z_2,\ldots,z_{k+1}). & \end{array}$$

Consider any pair $T_f(y,y_2,\ldots,y_\ell,z_{\ell+1},\ldots,z_{k+1}) + T_f(y,y_2,\ldots,y_{\ell-1},z_\ell,\ldots,z_{k+1})$ that appears in the above sum. Note that $T_f(y,y_2,\ldots,y_\ell,z_{\ell+1},\ldots,z_{k+1})$ and $T_f(y,y_2,\ldots,y_{\ell-1},z_\ell,\ldots,z_{k+1})$ differ only in a single parameter. Since $T_f(\cdot)$ is a symmetric function we can apply Claim 4 and obtain that

$$\begin{aligned} &T_f(y,y_2,\ldots,y_\ell,z_{\ell+1},\ldots,z_{k+1}) + T_f(y,y_2,\ldots,y_{\ell-1},z_\ell,\ldots,z_{k+1}) \\ &\quad = T_f(y+y_\ell,y_2,\ldots,y_{\ell-1},z_{\ell+1},\ldots,z_{k+1},y+y_\ell+z_\ell) \\ &\qquad\quad + T_f(y+z_\ell,y_2,\ldots,y_{\ell-1},z_{\ell+1},\ldots,z_{k+1},y+y_\ell+z_\ell) \end{aligned} \tag{10}$$

Recall that y is fixed and $y_2,\ldots,y_{k+1},z_2,\ldots,z_{k+1} \in \{0,1\}^n$ are uniformly selected, and so all parameters on the right hand side in the above equation are uniformly distributed. Also recall that by the definition of η, for $T_f(r_1,\ldots,r_{k+1})$, where r_i are uniformly selected at random, $\Pr_{r_1,\ldots,r_{k+1}\in\{0,1\}^n}[T_f(r_1,\ldots,r_{k+1}) \neq 0] = \eta$. Hence, by the union bound:

$$\begin{aligned}\delta &= \Pr_{y_2,\ldots,y_{k+1},z_2,\ldots,z_{k+1}\in\{0,1\}^n}[T_f(y,y_2,\ldots,y_{k+1}) + T_f(y,z_2,\ldots,z_{k+1}) = 0] \\ &\geq 1-2k\eta. \end{aligned} \tag{11}$$

By combining Equations (9) and (11) we get that $\gamma^2 + (1-\gamma)^2 \geq 1-2k\eta$. Since $\gamma \geq 1/2$ it follows that $\gamma = \gamma^2 + \gamma(1-\gamma) \geq \gamma^2 + (1-\gamma)^2 \geq 1-2k\eta$. $\square$

Lemma 5 *If $\eta < \frac{1}{(4k+2)2^k}$, then the function g belongs to $\mathcal{P}_k$.*

Proof. By Claim 1 it suffices to prove that if $\eta < \frac{1}{(4k+2)2^k}$, then $T_g(y_1, \ldots, y_{k+1}) = 0$, for every $y_1, \ldots, y_{k+1} \in \{0,1\}^n$. Let us fix the choice of $y_1, \ldots, y_{k+1}$, and recall that as defined in Equation (2), $T_g(y_1, \ldots, y_{k+1}) = \sum_{\emptyset \neq I \subseteq [k+1]} g(\sum_{i \in I} y_i)$. Suppose we uniformly select $k \cdot (k+1)$ random vectors $z_{i,j} \in \{0,1\}^n$, $1 \leq i \leq k+1$, $1 \leq j \leq k$. Then by Lemma 3, for every I, $\emptyset \neq I \subseteq [k+1]$, with probability at least $1 - 2k\eta$ over the choice of the $z_{i,j}$'s,

$$g\left(\sum_{i \in I} y_i\right) = T_f\left(\sum_{i \in I} y_i, \sum_{i \in I} z_{i,1}, \sum_{i \in I} z_{i,2}, \ldots, \sum_{i \in I} z_{i,k}\right) + f\left(\sum_{i \in I} y_i\right). \tag{12}$$

Let E_1 be the event that Equation (12) holds for all $\emptyset \neq I \subseteq [k+1]$. By the union bound:

$$\Pr[E_1] \geq 1 - (2^{k+1} - 1) \cdot 2k\eta \tag{13}$$

Assume that E_1 holds. Then

$$\begin{aligned}
&T_g(y_1, \ldots, y_{k+1}) \\
&\quad = \sum_{\emptyset \neq I \subseteq [k+1]} \left[T_f\left(\sum_{i \in I} y_i, \sum_{i \in I} z_{i,1}, \sum_{i \in I} z_{i,2}, \ldots, \sum_{i \in I} z_{i,k}\right) + f\left(\sum_{i \in I} y_i\right)\right] \\
&\quad = \sum_{\emptyset \neq I \subseteq [k+1]} \sum_{\emptyset \neq J \subseteq [k]} \left[f\left(\sum_{i \in I} \sum_{j \in J} z_{i,j}\right) + f\left(\sum_{i \in I} y_i + \sum_{i \in I} \sum_{j \in J} z_{i,j}\right)\right] \\
&\quad = \sum_{\emptyset \neq J \subseteq [k]} \sum_{\emptyset \neq I \subseteq [k+1]} f\left(\sum_{i \in I} \sum_{j \in J} z_{i,j}\right) \\
&\qquad + \sum_{\emptyset \neq J \subseteq [k]} \sum_{\emptyset \neq I \subseteq [k+1]} f\left(\sum_{i \in I} y_i + \sum_{i \in I} \sum_{j \in J} z_{i,j}\right) \\
&\quad = \sum_{\emptyset \neq J \subseteq [k]} T_f\left(\sum_{j \in J} z_{1,j}, \ldots, \sum_{j \in J} z_{k+1,j}\right) \\
&\qquad + \sum_{\emptyset \neq J \subseteq [k]} T_f\left(y_1 + \sum_{j \in J} z_{1,j}, \ldots, y_{k+1} + \sum_{j \in J} z_{k+1,j}\right).
\end{aligned} \tag{14}$$

Let E_2 be the event that for every $\emptyset \neq J \subseteq [k]$, $T_f\left(\sum_{j \in J} z_{1,j}, \ldots, \sum_{j \in J} z_{k+1,j}\right) = 0$ and $T_f\left(y_1 + \sum_{j \in J} z_{1,j}, \ldots, y_{k+1} + \sum_{j \in J} z_{k+1,j}\right) = 0$. By the definition of η:

$$\Pr[E_2] \geq 1 - 2(2^k - 1)\eta \tag{15}$$

Suppose that $\eta < \frac{1}{(4k+2)2^k}$. Then, by Equations (13) and (15), the probability that both E_1 and E_2 hold, is strictly positive. In other words, there exists a choice of the $z_{i,j}$'s for which all summands in Equation (14) are 0. But this implies that $T_g(y_1, \ldots, y_{k+1}) = 0$. We conclude that if $\eta < \frac{1}{(4k+2)2^k}$, then g belongs to $\mathcal{P}_k$, and this completes the lemma's proof.

By combining Lemmas 2 and 5 we obtain that if f is $\Omega(1/(k2^k))$-far from $\mathcal{P}_k$, then $\eta = \Omega(1/(k2^k))$, and so the algorithm rejects f with sufficiently high constant probability (since it selects $\Omega(k2^k)$ groups of vectors $y_1, \ldots, y_{k+1}$). We next deal with the case in which η is small. By Lemma 2, in this case the distance $d = \mathrm{dist}(f, g)$ between f and g is small, and we show that the test rejects f with probability that is close to $(2^{k+1} - 1)d$. This follows from the fact that in this case, the probability over the selection of $y_1, \ldots, y_{k+1}$, that among the $(2^{k+1} - 1)$ points $\sum_{\emptyset \neq I \subseteq [k+1]} y_i$, the functions f and g differ in precisely one point, is close to $(2^{k+1} - 1)d$. This is formally proved in the following lemma.

Lemma 6 *Suppose* $0 < \eta < \frac{1}{(4k+2)2^k}$. *Let* $d = \mathrm{dist}(f, g)$ *denote the distance between* f *and* g, *and let*

$$p \stackrel{\mathrm{def}}{=} \frac{1 - (2^{k+1} - 1)d}{1 + (2^{k+1} - 1)d} \cdot (2^{k+1} - 1)d.$$

Then, when $y_1, y_2, \ldots, y_{k+1}$ *are chosen randomly, the probability that for exactly one point* v *among the* $(2^{k+1} - 1)$ *points* $\sum_{i \in S} y_i$, $(\emptyset \neq S \subseteq [k+1])$, $f(v) \neq g(v)$, *is at least* p.

By definition of η and the above lemma, $\eta \geq p$ (under the premise of the lemma). In particular, since (by Lemma 2) $d \leq 2\eta \leq \frac{1}{(2k+1)2^k}$ and $k \geq 1$, $\eta \geq \frac{1}{3}(2^{k+1} - 1)d$, and, for fixed k, as d tends to zero, $\eta \geq (2^{k+1} - 1)d - O(d^2)$.

Proof. For each subset S, $\emptyset \neq S \subseteq [k+1]$, let X_S be the indicator random variable whose value is 1 if and only if $f(\sum_{i \in S} y_i) \neq g(\sum_{i \in S} y_i)$. Obviously, $\Pr[X_S = 1] = d$ for every S. It is not difficult to check that the random variables X_S are pairwise independent, since for any two distinct nonempty S_1, S_2, the sums $\sum_{i \in S_1} y_i$ and $\sum_{i \in S_2} y_i$ attain each pair of distinct values in $\{0, 1\}^n$ with equal probability when the vectors y_i are chosen randomly and independently. It follows that the random variable $X = \sum_S X_S$ which counts the number of points v of the required form in which $f(v) \neq g(v)$ has expectation $\mathrm{E}[X] = (2^{k+1} - 1)d$ and variance $\mathrm{Var}[X] = (2^{k+1} - 1)d(1 - d) \leq \mathrm{E}[X]$. Our objective is to lower bound the probability that $X = 1$. We need the well known, simple fact that for a random variable X that attains nonnegative, integer values,

$$\Pr[X > 0] \geq \frac{(\mathrm{E}[X])^2}{\mathrm{E}[X^2]}.$$

Indeed, if X attains the value i with probability p_i for $i > 0$, then, by Cauchy-Schwartz,

$$(\mathrm{E}[X])^2 = (\sum_{i>0} i p_i)^2 = (\sum_{i>0} i \sqrt{p_i} \sqrt{p_i})^2 \leq (\sum_{i>0} i^2 p_i)(\sum_{i>0} p_i) = \mathrm{E}[X^2] \Pr[X > 0].$$

In our case, this implies

$$\Pr[X > 0] \geq \frac{(\mathrm{E}[X])^2}{\mathrm{E}[X^2]} \geq \frac{(\mathrm{E}[X])^2}{\mathrm{E}[X] + (\mathrm{E}[X])^2} = \frac{\mathrm{E}[X]}{1 + \mathrm{E}[X]}.$$

Therefore

$$\mathrm{E}[X] \geq \Pr[X = 1] + \left(\frac{\mathrm{E}[X]}{1 + \mathrm{E}[X]} - \Pr[X = 1]\right) \cdot 2 = \frac{2\mathrm{E}[X]}{1 + \mathrm{E}[X]} - \Pr[X = 1],$$

implying that

$$\Pr[X = 1] \geq \frac{\mathrm{E}[X] - (\mathrm{E}[X])^2}{1 + \mathrm{E}[X]}.$$

Substituting the value of $\mathrm{E}[X]$, the desired result follows.

We are now ready to wrap-up the proof of Theorem 1.
Proof of Theorem 1: As we have noted previously, if f is in $\mathcal{P}_k$, then by Claim 1 the tester accepts (with probability 1). We next show that if f is ϵ-far from $\mathcal{P}_k$, then the tester rejects with probability at least $\frac{2}{3}$.
Suppose that $dist(f, \mathcal{P}_k) > \epsilon$. Denote $d = dist(f, g)$. If $\eta < \frac{1}{(4k+2)2^k}$ then by Lemma 5 $g \in \mathcal{P}_k$ and, by Lemma 6, $\eta \geq \Omega(2^k d) \geq \Omega(2^k \epsilon)$. Hence, $\eta \geq min\left(\Omega(2^k\epsilon), \frac{1}{(4k+2)2^k}\right)$. Clearly it is enough to perform $O(\frac{1}{\eta})$ rounds of the algorithm in order to detect a violation with probability at least $\frac{2}{3}$. This completes the proof of the theorem. □

4.1 Self-Correcting and a Lower Bound

From Lemmas 2, 3, and 5 one can immediately conclude the following:

Corollary 7 *Consider a function $f : \{0,1\}^n \to \{0,1\}$ that is ϵ-close to a degree-k polynomial $g : \{0,1\}^n \to \{0,1\}$, where $\epsilon < \frac{2}{(4k+2)2^k}$. Then the function f can be self-corrected. That is, for any given $x \in \{0,1\}^n$, it is possible to obtain the value $g(x)$ with probability at least $1 - \epsilon k$ by querying f on $2^k - 1$ points in $\{0,1\}^n$.*

The following is a lower bound on families of functions that correspond to linear codes.

Theorem 2 *Let $\mathcal{F}$ be any family of functions $f : \{0,1\}^n \to \{0,1\}$ that corresponds to a linear code $\mathcal{C}$. Let d denote the minimum distance of the code $\mathcal{C}$ and let $\bar{d}$ denote the minimum distance of the dual code of $\mathcal{C}$.*
Every testing algorithm for the family $\mathcal{F}$ must perform $\Omega(\bar{d})$ queries, and if the distance parameter ϵ is at most $d/2^{n+1}$, then $\Omega(1/\epsilon)$ is also a lower bound for the necessary number of queries.

As noted in the introduction, the family $\mathcal{P}_k$ corresponds to the shortened Reed-Muller code $\mathcal{R}(k,n)^*$. It is well known (see [16, Chap. 13]) that the distance of $\mathcal{R}(k,n)^*$ is 2^{n-k} and the distance of the dual code (which is a punctured Reed-Muller code) is $2^{k+1} - 1$. Hence we obtain the following corollary.

Corollary 8 *Every algorithm for testing $\mathcal{P}_k$ with distance parameter ϵ must perform $\Omega\left(\max(\frac{1}{\epsilon}, 2^{k+1})\right)$ queries.*

Proof of Theorem 2: We start with showing that $\Omega(\bar{d})$ queries are necessary. A well known fact from coding theory (see [16, Chap. 5]) states the following: for every linear code $\mathcal{C}$ whose dual code has distance $\bar{d}$, if we examine a sub-word having length d', $d' < \bar{d}$, of a uniformly selected codeword in $\mathcal{C}$, then the resulting sub-word is uniformly distributed in $\{0,1\}^{d'}$. Hence it is not possible to distinguish between a random codeword in $\mathcal{C}$ and a random word in 2^n (which with high probability is far from any codeword) using less than $\bar{d}$ queries.
We now turn to the case $\epsilon < d/2^{n+1}$. To prove the lower bound here, we apply, as usual, the Yao principle by defining two distributions, one of positive instances, and the other of negative ones, and then by showing that in order to distinguish between those distributions any algorithm must perform $\Omega(1/\epsilon)$ queries. The positive distribution has all its mass at the zero vector $\bar{0} = (0,\ldots,0)$. To define the negative distribution, partition the set of all coordinates into $t = 1/\epsilon$ nearly equal parts $I_1,\ldots,I_t$ and give weight $1/t$ to each of the characteristic vectors w_i of I_i, $i = 1,\ldots,t$. (Observe that indeed $\bar{0} \in \mathcal{C}$ due to linearity, and $dist(w_i, \mathcal{C}) = \epsilon$ due to the assumption on the minimum distance of $\mathcal{C}$). Finally, a random instance is generated by first choosing one of the distributions with probability $1/2$, and then generating a vector according to the chosen distribution. It is easy to check (see, e.g., [1] for details) that in order to give a correct answer with probability at least $2/3$, the algorithm has to query $\Omega(1/\epsilon)$ bits of the input.
□

5 Concluding remarks

We first note that in view of the above lower bound, our upper bound is almost tight.
It will be interesting to study analogous questions for other linear binary codes. Several recent papers, including [8], [9], deal with related questions. As shown above, a code is not testable with a constant number of queries if its dual distance is not a constant, and it seems plausible to conjecture that if the dual distance is a constant, and there is a doubly transitive permutation group acting on the coordinates that maps the dual code to itself, then the code can be testable with a constant number of queries. The automorphism group of punctured Reed-Muller codes contains the general linear group GL$(n,2)$, and thus those codes supply an example with these properties. Another interesting example is duals of BCH codes (this class also contains linear functions as a particular case). Another possible extension of the results could be the study of testability of low-degree multivariate polynomials over small fields $GF(q)$. This situation corresponds to generalized Reed-Muller codes [15].

References

1. N. Alon, M. Krivelevich, I. Newman, and M. Szegedy. Regular languages are testable with a constant number of queries. In *Proceedings of the Fortieth Annual Symposium on Foundations of Computer Science*, pages 645–655, 1999.

2. S. Arora and S. Safra. Improved low-degree testing and its applications. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing*, pages 485–495, 1997.
3. L. Babai, L. Fortnow, L. Levin, and M. Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*, pages 21–31, 1991.
4. L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1(1):3–40, 1991.
5. M. Bellare, D. Coppersmith, J. Håstad, M. Kiwi, and M. Sudan. Linearity testing in characteristic two. In *Proceedings of the Thirty-Sixth Annual Symposium on Foundations of Computer Science*, pages 432–441, 1995.
6. M. Bellare, S. Goldwasser, C. Lund, and A. Russell. Efficient probabilistically checkable proofs and applications to approximation. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on the Theory of Computing*, pages 294–304, 1993.
7. M. Bellare and M. Sudan. Improved non-approximability results. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on the Theory of Computing*, pages 184–193, 1994.
8. E. Ben-Sasson, P. Harsha, and S. Raskhodnikova. 3CNF properties are hard to test. In *Proceedings of the Thirty-Fifth Annual ACM Symposium on the Theory of Computing*, 2003. To appear.
9. E. Ben-Sasson, M. Sudan, S. Vadhan, and A. Wigderson. Derandomizing low degree tests via epsilon-biased spaces. In *Proceedings of the Thirty-Fifth Annual ACM Symposium on the Theory of Computing*, 2003. To appear.
10. M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47:549–595, 1993.
11. U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Approximating clique is almost NP-complete. *Journal of the Association for Computing Machinery*, pages 268–292, 1996.
12. K. Friedl and M. Sudan. Some improvements to total degree tests. In *Proceedings of the 3rd Annual Israel Symposium on Theory of Computing and Systems*, pages 190–198, 1995. Corrected version available online at http://theory.lcs.mit.edu/˜madhu/papers/friedl.ps.
13. P. Gemmell, R. Lipton, R. Rubinfeld, M. Sudan, and A. Wigderson. Self-testing/correcting for polynomials and for approximate functions. In *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*, pages 32–42, 1991.
14. M. Hall. *Combinatorial Theory*. John Wiley & Sons, 1967.
15. T. Kasami, S. Lin, and W.W. Peterson. New generalizations of the reed-muller codes, part i: Primitive codes. *IEEE Transactions on Information Theory*, pages 189–199, 1968.
16. F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error Correcting Codes*. North Holland, 1977.
17. R. Rubinfeld and M. Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996.
18. M. Sudan. Private communications, 1995.