Limits of Approximation Algorithms 12 Mar, 2010 (IMSc)

Lec. 6: Linearity Testing over $\mathbb{GF}(2)$ via Fourier Analysis

Lecturer: Prahladh Harsha *Scribe: Yadu Vasudev*

The agenda for today's lecture is

- Analysis of Linearity testing using Fourier technique.
- Introduction to Coding theory and polynomial codes.
- Proof of CIRCUIT-SAT $\in \mathsf{PCP}_{1,1-\varepsilon}(O(n^2), O(1))$

The references for this lecture include Lectures 10 and 13 from Sudan's course on inapproximability at MIT [Sud99] and Lectures 3 and 4 from a course on PCPs at Univ of Chicago [Har07]. Parts of the these notes have been adapted from the notes scribed by Joshua A.Grochow (Lecture 3) and Andrew Cotter (Lecture 4) for the Univ. of Chicago course by the same instructor.

6.1 Linearity Testing and Fourier Analysis

Recall the BLR test for linearity from last lecture:

$$\begin{aligned}\mathsf{BLR\text{-}Test}(f):\ &1.\ \text{Choose } x, y \in_R G\\ &2.\ \text{Accept if } f(x)+f(y)=f(x+y).\end{aligned}$$

Randomness of BLR-Test: In this test, the number of random bits used is $2\log|G|$. Since there are $|G|$ elements to be considered, atleast $\log|G|$ number of random bits seem necessary. Do we really require an additional $\log|G|$ random coins. We could ask if the BLR-test can be done using just $\log|G| + o(\log|G|)$ random bits and $O(1)$ queries. Shpilka and Wigderson [SW06] show that this is in fact possible, if one chooses the two points x and y as vertices of a random edge in a Cayley graph on G instead of independently. The error probability of the test in this case depends on the second eigenvalue of the Cayley graph. For a group G and a set of generators $S = \{s_1, \dots, s_k\}$ for G closed under negation (i.e., $s \in S$ implies $-s \in S$) the corresponding Cayley graph $C(G,S) = (V,E)$ is constructed as follows. $V = G$ and $E = \{(x, x+s) | x \in V, s \in S\}$.

6.1.1 Fourier Analysis

In this section, we give better bounds for the linearity test of [BLR93] over $\mathbb{GF}(2)$ instead of general groups, using Fourier analysis. This analysis is due to [BCH$^+$96].

We now look at functions $f : G \to H$, where $G = \{0,1\}^n$ and $H = \{0,1\}$. Any linear function is of the form

$$l_\alpha(X) = \sum_{i=1}^{n} \alpha_i x_i,\ \alpha \in \{0,1\}^n$$

It will be more convenient if we map $0 \to 1$ and $1 \to -1$ Hence, from now on we look at functions $f : \{0,1\}^n \to \{1,-1\}$.. In this new notation, we write

$$\begin{aligned}\chi_\alpha(X) &= (-1)^{l_\alpha(X)} \\ &= (-1)^{\sum_{i=1}^{n} \alpha_i x_i}\end{aligned}$$

Consider the set $\mathcal{F} = \{f : \{0,1\}^n \to \mathbb{R}\}$. This set forms a vector space of dimension 2^n, where addition is defined as $(f+g)(x) = f(x) + g(x)$. We will show that the set $\{\chi_\alpha : \alpha \in \{0,1\}^n\}$ forms an orthonormal basis under a suitably defined inner-product for $\mathcal{F}$ and this basis will be useful in analysing the BLR-test. To that end, define an inner product on $\mathcal{F}$ as follows:

$$\langle f, g\rangle = \mathbb{E}_{x\in\{0,1\}^n}\left[f(x)g(x)\right] = \frac{1}{2^n} \cdot \sum_{x\in\{0,1\}^n} f(x)g(x).$$

Observation 6.1.1. $\mathbb{E}_x[\chi_\alpha(x)] = \begin{cases} 1 & \text{if } \alpha = 0 \\ 0 & \text{otherwise}\end{cases}$

Observation 6.1.2. $\mathbb{E}_x[\chi_\alpha(x)\cdot\chi_\beta(x)] = \mathbb{E}_x[\chi_{\alpha+\beta}(x)] = \begin{cases} 1 & \text{if } \alpha = \beta \\ 0 & \text{otherwise}\end{cases}$

Thus $\{\chi_\alpha(x) | \alpha \in \{0,1\}^n\}$ are an orthonormal set of vectors, and since there are 2^n many of them they form an orthonormal basis for $\mathcal{F}$. Thus for any $f \in \mathcal{F}$, we have

$$f = \sum \hat{f}_\alpha \chi_\alpha,$$

where $\hat{f}_\alpha = \langle f, \chi_\alpha\rangle$ are known as the Fourier coefficients of f. This is called the *Fourier representation* of f. Observe that any linear function puts all its weight on the corresponding Fourier coefficient other functions are written as linear combinations of the linear functions.

We have the following identities about the Fourier coefficients

Observation 6.1.3 (Plancherel's Identity)**.** *For $f, g \in \mathcal{F}$, $\langle f, g\rangle = \sum \hat{f}_\alpha \cdot \hat{g}_\alpha$*

Proof. We know

$$\begin{aligned}\langle f, g\rangle &= \langle \sum_\alpha \hat{f}_\alpha\chi_\alpha, \sum_\alpha \hat{g}_\alpha\chi_\alpha\rangle \\ &= \sum_\alpha\sum_\beta \hat{f}_\alpha\hat{g}_\beta\langle\chi_\alpha,\chi_\beta\rangle \text{ (by linearity of the inner product)} \\ &= \sum \hat{f}_\alpha\cdot\hat{g}_\alpha \text{ (by the orthonormality of } \chi_\alpha\text{'s)}\end{aligned}$$

□

which gives us the following two corollaries

Corollary 6.1.4 (Parseval's Identity)**.** *For $f \in \mathcal{F}$, $\|f\|^2 = \langle f, f\rangle = \sum_\alpha \hat{f}_\alpha^2$*

Corollary 6.1.5. *If $f : \{0,1\}^n \to \{1,-1\}$, then $\|f\|^2 = \langle f, f\rangle = 1$*

6.1.2 Analysis of the Linearity test

Recall the definitions,

$$\begin{aligned}
\varepsilon(f) &= \Pr_{x,y}[f(x+y) \neq f(x) + f(y)] \\
\delta(f) &= \delta(f, linear) \\
&= \min_{\alpha} \delta(f, \chi_\alpha) \\
&= \min_{\alpha} \ \Pr_{x}[f(x) \neq \chi_\alpha(x)]
\end{aligned}$$

Coppersmith's analysis for the BLR-test on general groups showed that if $\varepsilon(f) < \frac{2}{9}$ then $\delta(f) \leq 2\varepsilon(f)$. Using Fourier analysis, this can be tightened to show the following theorem when $G = \{0,1\}^n$.

Theorem 6.1.6. *For $f : \{0,1\}^n \to \{1,-1\}$, $\delta(f) \leq \varepsilon(f)$*

This bound seems to be optimum because if f differs from every linear function in atleast δ values, the error probability of the BLR-test should be atleast so much. Before proving the theorem we have the following lemma, which shows that f is closest to the linear function on which it has the largest component.

Lemma 6.1.7. *For $f : \{0,1\}^n \to \{1,-1\}$, $\delta(f) = \frac{1}{2}(1 - \max_\alpha \ \hat{f}_\alpha)$*

Proof. This follows from the definition of $\delta(f)$ and the Fourier coefficients

$$\begin{aligned}
\delta(f) &= \min_{\alpha}\Big\{\Pr_{x}[f(x) \neq \chi_\alpha(x)]\Big\} \\
&= \min_{\alpha}\Big\{\mathbb{E}_x\Big[\frac{1 - f(x)\chi_\alpha(x)}{2}\Big]\Big\} \\
&= \frac{1}{2}(1 - \max_{\alpha}\{\langle f, \chi_\alpha\rangle\}) \\
&= \frac{1}{2}(1 - \max_{\alpha}\{\hat{f}_\alpha\})
\end{aligned}$$

□

Proof of Theorem 6.1.6.

$$\begin{aligned}
\varepsilon(f) &= \mathbb{E}_{x,y}\Big[\frac{1 - f(x+y)f(x)f(y)}{2}\Big] \\
&= \frac{1}{2} - \frac{1}{2}\mathbb{E}_{x,y}[f(x+y)f(x)f(y)]
\end{aligned}$$

Let's look at $\mathbb{E}_{x,y}[f(x+y)f(x)f(y)]$.

$$\begin{aligned}
\mathbb{E}_{x,y}[f(x+y)f(x)f(y)] &= \mathbb{E}_{x,y}\Big[\Big(\sum_{\alpha}\hat{f}_\alpha\chi_\alpha(x+y)\Big)\Big(\sum_{\beta}\hat{f}_\alpha\chi_\beta(x)\Big)\Big(\sum_{\gamma}\hat{f}_\alpha\chi_\gamma(y)\Big)\Big] \\
&= \mathbb{E}_{x,y}\Big[\sum_{\alpha,\beta,\gamma}\hat{f}_\alpha\hat{f}_\beta\hat{f}_\gamma\chi_\alpha(x+y)\chi_\beta(x)\chi_\gamma(y)\Big] \\
&= \sum_{\alpha,\beta,\gamma}\hat{f}_\alpha\hat{f}_\beta\hat{f}_\gamma\mathbb{E}_x\left[\chi_\alpha(x)\chi_\beta(x)\right]\mathbb{E}_y\left[\chi_\alpha(y)\chi_\beta(y)\right] \\
&= \sum_{\alpha,\beta,\gamma}\hat{f}_\alpha\hat{f}_\beta\hat{f}_\gamma\langle\chi_\alpha,\chi_\beta\rangle\langle\chi_\alpha,\chi_\gamma\rangle \\
&= \sum_{\alpha}\hat{f}_\alpha^3
\end{aligned}$$

Thus we have

$$\begin{aligned}
\varepsilon(f) &= \frac{1}{2}-\frac{1}{2}\sum_{\alpha}\hat{f}_\alpha^3 \\
&\geq \frac{1}{2}-\frac{1}{2}(\max_{\alpha}\hat{f}_\alpha)\sum_{\alpha}\hat{f}_\alpha^2 \\
&= \frac{1}{2}-\frac{1}{2}\max_{\alpha}\hat{f}_\alpha = \delta(f)
\end{aligned}$$

□

Improving the amortized query complexity: For any function f, let $d(f)$ denote $\max_\alpha \hat{f}_\alpha$. Hence, the maximum agreement of f with a linear function is $(1+d(f))/2$. The above analysis shows that the error $\varepsilon(f)$ of the BLR-Test in this case is at most $(1+d(f))/2$. Can we reduce this error? Clearly, if we repeat the BLR-test k times, and accept iff the test accepts each time, then the probability that it accepts a function that is far from linear drops exponentially in k. In fact, it can easily be shown that the error in this case is $(1/2+d(f)/2)^k$. The k-repeated BLR-Test makes $3k$ queries. Hence, the error drops roughly by a factor of $1/2$ for every additional 3 queries. Can we improve this rate of fall in error with each additional query (in the amortized sense). This quantity is called the amortized query complexity (i.e, if a q query test has error at most ε and perfect completeness, then the amortized query complexity is $q/log_2(1/\varepsilon)$.) This question was studied by Samorodnitsky, Sudan and Trevisan and finally Håstad and Wigderson [HW03] showed the following: Consider the following test. Pick $x_1,\ldots,x_t \in_R \{0,1\}^n$. For each pair $(i,j)\in[t]\times[t]$ run the BLR-test, i.e., check if $f(x_i)+f(x_j)=f(x_i+x_j)$. It can be shown that the error of this test is at most $2^{-\binom{t}{2}}+d(f)$ while it makes just $t+\binom{t}{2}$ queries. Thus, the amortized query complexity of this test is $1+O(1/\sqrt{q})$ (Here $q=t+\binom{t}{2}$). The analysis of this test is similar to the above analysis for the simple test. An extended version of this test is used to improve the amortized query complexity of PCPs which in turn leads to improved inapproximability results for MAXCLIQUE.

6.2 Introduction to Coding theory and Polynomial codes

A PCP is a rewritting of the NP proof such that the new proof is checkable. For this purpose, we first need codes which are locally checkable (also called locally testable). We will now see a short introduction to coding theory, consider different types of codes and their checkable properties.

6.2.1 Codes Primer

The basic aim in coding theory, is to convert a message M into a new one $E(\mathsf{M})$, such that even after passing through a channel with noise η, M can be retrieved from $E(\mathsf{M})$ by the reciever.
A code is a function $\mathcal{C} : \Sigma^k \to \Sigma^n$, which converts a k-symbol message to an n-symbol codeword.
We will use the following notation, henceforth. The size of the alphabet $|\Sigma| = q$, k denotes the message length, n the block length. $\frac{k}{n}$ is known as the rate of the code, which measures the amount of information per symbol of the codeword. The symbol $\mathcal{C}$ will also be used for the set of codewords $\mathcal{C}(\Sigma^k)$.

Definition 6.2.1. *The distance between 2 codewords x and y is defined as $d(x, y) = \big|\{i : x_i \neq y_i\}\big|$. The distance of a code $\mathcal{C}$ is defined as $d(\mathcal{C}) = \min_{x \neq y} d(x, y)$. Thus it is the shortest distance between any two distinct codewords.*

Thus, if the noise η added by the channel is at most $\frac{d(\mathcal{C})}{2}$, then one can uniquely identify (at least existentially) the codeword closest to the corrupted word. We will denote $\mathcal{C} : \Sigma^k \to \Sigma^n$ with distance d as a $(n, k, d)_\Sigma$-code. If Σ is a finite field $\mathbb{F}(|\mathbb{F}| = q)$ and $\mathcal{C}$ is a vector space over $\mathbb{F}$, then $\mathcal{C}$ is known as a linear code and denoted by $[n, k, d]_q$-code.

From an algorithmic perspective, the following questions are interesting for a code $\mathcal{C} : \Sigma^k \to \Sigma^n$

Encoding: Given $m \in \Sigma^k$, compute $\mathcal{C}(m)$.

Testing: Given $w \in \Sigma^n$, check if w is a valid codeword.

Decoding: Given $\mathcal{C}(m) + \eta$ s.t $|\eta| < \frac{d(\mathcal{C})}{2}$, compute m.

In some cases, like in the context of PCPs, we are also interested in the sub-linear time equivalent of these problems, (i.e, algorithms which can compute the above results even without reading the entire input string, but only querying it at a few locations). It is unlikely that encoding will have a sub-linear time algorithm since a codeword should ideally depend on all the symbols in the message. A code which has a sub-linear time algorithm for testing is known as a *locally-testable* code and similarly, a code with a sublinear-time decoding procedure is said to be *locally-decodable.* We will not define these terms formally.

Walsh-Hadamard code

An example of a linear code is the Walsh-Hadamard (WH) code. It is defined over the field $\mathbb{F}_2$. Given any $x \in \{0, 1\}^n$, the Walsh-Hadamard encoding of x is a string of size 2^n

where the α^{th}-bit corresponds to the function $l_\alpha(x) = \sum_i \alpha_i x_i, \alpha \in \{0,1\}^n$. More precisely, $\mathsf{WH}(x)_\alpha = l_\alpha(x), \forall \alpha \in \{0,1\}^n$. It follows from the properties of linear functions (specifically Observation 6.1.2) that the distance between any two distinct WH codewords is 2^{n-1}. Thus, the WH code is a $[k, 2^k, 2^{k-1}]_2$ code. The linearity testing algorithm shows that it is locally-testable. A local decoder for Walsh-Hadamard code is as follows. For any function $f : \{0,1\}^n \to \{0,1\}$, that is supposedly close to some linear function l_z, the local decoder $\mathsf{Dec}(f) : \{0,1\}^n \to \{0,1\}$ is defined as follows:

```
Dec(f) :On input x,
    1. Choose $r \in_R \{0,1\}^n$
    2. Output $f(x+r) - f(r)$
```

If f is δ-close to the Walsh-Hadamard code l_z for some $z \in \{0,1\}^n$, then

$$\Pr[\mathsf{Dec}(f) \text{ decodes correctly}] \geq 1 - 2\delta.$$

Thus, the code has wonderful distance, local testability and local decodability properites. The main weakness of this code is its poor rate, it blows-up the size of the message exponentially. In the next section, we will use the Walsh-Hadamard code and linearity testing to construct an exponential sized PCP.

For applications in constructing good (i.e, polynomial sized) PCPs, we require codes which have a small blow-up (at most polynomially) and still have good distance and testability properties. In the next section, we will see codes with inverse polynomial rate and good distance, testability and decodability (though not as good as the WH code in the latter properties).

6.2.2 Polynomial Codes

In this section, we will look at codes in which the underlying alphabet Σ is some finite field, say $\mathbb{F}$. We will construct codes using low-degree univariate polynomials (of the form $p(x) = \sum_{i=0}^d a_i x^i$) and low-degree multivariate polynomials (of the form $p(x_1, \ldots, x_n) = \sum_{e_1+\cdots+e_n \leq d} a_{e_1,\ldots,e_n} x_1^{e_1} \ldots x_n^{e_n}$). The following property of univariate polynomials will come very useful.

Observation 6.2.2. *If p is a univariate polynomial of degree d that is not identically zero, then p has atmost d roots in $\mathbb{F}$.*

This observation can be extended to the case of multivariate polynomials p that are not identically zero as follows.

Lemma 6.2.3 (Schwartz-Zippel Lemma [Sch80, Zip79])**.** *Let p be a multivariate polynomial of degree d that is not identically zero,*

$$\Pr_{X \in \mathbb{F}^n}[p(X) = 0] \leq \frac{d}{|\mathbb{F}|}$$

Reed-Solomon codes

For a finite field $\mathbb{F}$, the Reed-Solomon code (RS) is a function $\mathsf{RS} : \mathbb{F}^k \to \mathbb{F}^{|\mathbb{F}|}$. A message string $a_0, \ldots, a_k$ is mapped to the polynomial $p(x) = \sum_{i=0}^{k} a_i x^i$ and $\mathsf{RS}(a_0, \ldots, a_k) = (p(f_1), \ldots, p(f_{|\mathbb{F}|}))$, where f_is are all the elements of the field.

Clearly, this is a $[k, n, n-k-1]_n$ code where n is both the size of the field and the block-length. These codes are not a good choice for constructing PCPs since they are "strongly" non-locally testable, i.e at least k queries (equal to the length of the original message) have to be made to the codeword.

Reed-Muller Codes

The Reed-Muller(RM) codes is a function $\mathsf{RM} : \mathbb{F}^{\binom{n+d}{d}} \to \mathbb{F}^{|\mathbb{F}|^n}$. The message can be looked as the coefficients of a polynomial of degree d, for a suitable d. The codeword is the evaluation of the polynomial at all the $\mathbb{F}^n$ points. The Schwartz-Zippel lemma gives the fact that two different polynomials evaluate to the same value in atmost $\frac{d}{|\mathbb{F}|}$ positions. Thus RM is a $[\binom{n+d}{d}, q^n, (1-\frac{d}{q})q^n]_q$ code, where $q = |\mathbb{F}|$.

The Reed-Muller codes have good rates and the local-testability property. The local testing of these codes will be the topic of the next lecture.

6.3 PCP from WH codes and Linearity testing

We now give use the linearity test to construct an exponential sized PCP, and thus, show that CIRCUIT-SAT $\in \mathsf{PCP}_{1,1-\varepsilon}(O(n^2), O(1))$. The decision problem of CIRCUIT-SAT is as follows: given a circuit $\mathcal{C}$ with gates labelled from 1 to n and each gate having fan-in 2, does there exists an input assignment on which the circuit evaluates to 1? CIRCUIT-SAT$\in$ NP since we can consider any assignment to the variables which cause the circuit to evaluate to 1 as a certificate. We will now give a constant query, albeit exponential sized PCP for CIRCUIT-SAT.

Any assignment (both satisfying and non-satisfying) to the gates of the circuit C can be viewed as a function $z : \{0,1\}^n \to \{0,1\}$. The PCP will include the Walsh-Hadamard encoding of a satisfying assignment. Local testability of WH codes imply that we can check if the given string is actually a valid codeword or not. But we also need to check if it is the encoding of a satsifying assignment, i.e., an assignment that satisfies all the gates (including the output gate). For this purpose, we can express the gate constraints as a quadratic function as follows (assume w.l.o.g that the circuit contains only AND and NOT gates). For each gate i with inputs j and k,

$$P_i(z) = \begin{cases} z_i - z_j z_k & \text{if the } i\text{-th gate is an AND gate with inputs from} \\ & \text{gates } j \text{ and } k. \\ z_i - (1 - z_j) & \text{if the } i\text{-th gate is a NOT gate with input from} \\ & \text{gate } j. \\ 1 - z_j & \text{if the } i\text{-th gate is an output gate with input from} \\ & \text{gate } j. \\ 0 & \text{if the } i\text{-th gate is an input gate (i.e. } i \le m). \end{cases}$$

Thus the question that we now ask is whether there exists an assignment z to the different gates such that $\forall$ gates $i, P_i(z) = 0$. Since the constraints for the gates includes quadratic constraints, the PCP proof, in addition to the Walsh-Hadamard encoding of the assignment z, will also include a quadratic equivalent of the Walsh-Hadamard codes. More precisely, the PCP proof for an assignment z is as follows

$$\mathsf{WH}(z) = \left\{ l_\alpha(z) \middle| \alpha \in \{0,1\}^n \right\}$$
$$\mathsf{Quad}(z) = \left\{ \sum_{i=1}^{n} \sum_{j=1}^{n} M_{ij} z_i z_j \middle| \forall M \in \{0,1\}^{n^2} \right\}$$

We now describe the verification of the PCP.

Assume that the proof consists of functions $A : \{0,1\}^n \to \{0,1\}$ and $B : \{0,1\}^{n^2} \to \{0,1\}$, the following tests are performed. (The functions $A(z)$ and $B(z)$ are supposedly $\mathsf{WH}(z)$ and $\mathsf{Quad}(z)$, but ofcourse, the verifier needs to check that this is indeed the case).

- *Codeword Test* : This test checks whether the function A and B are linear functions or far away from linear functions.
  - Pick $\alpha, \beta \in_R \{0,1\}^n$. Check if $A(\alpha) + A(\beta) = A(\alpha + \beta)$
  - Pick $M_1, M_2 \in_R \{0,1\}^{n \times n}$. Check if $B(M_1) + B(M_2) = B(M_1 + M_2)$.
- *Consistency Test* : This test aims to check whether the A and B are the linear and quadratic functions of the same assignment z. For $\alpha, \beta \in \{0,1\}^n$, consider the matrix M, where $M_{i,j} = \alpha_i \beta_j$. Thus

  $$\sum_{i=1}^{n} \sum_{j=1}^{n} M_{i,j} z_i z_j = \left(\sum_{i=1}^{n} \alpha_i z_i \right) \left(\sum_{i=1}^{n} \beta_i z_i \right)$$

  This suggest the following natural test: "Pick $\alpha, \beta \in_R \{0,1\}^n$ and let $M_{i,j} = \alpha_i \beta_j$. Check if $B(M) = A(\alpha)A(\beta)$." However, since we only know that A and B are close to being linear (if they are far, the *Codeword Test* rejects), we perform the following self-corrected version of the above test: "Pick $\alpha, \beta \in_R \{0,1\}^n$ and let $M_{i,j} = \alpha_i \beta_j$. Pick $N \in_R \{0,1\}^{n \times n}$. Check if $B(M + N) - B(N) = A(\alpha) \cdot A(\beta)$."
- *Circuit test* : We now need to check that z satisfies all the gates. Instead of testing this for all the gates which is very query intensive, we instead check if z satisfies a random linear combination of all the gates. More precisely, for each gate g pick $r_g \in_R \{0,1\}$ and consider the function $p = \sum_g r_g P_g$ which can be written as the sum of quadratic terms, linear forms and a constant. Let $p = \sum_{i=1}^{n} \sum_{j=1}^{n} M_{i,j} z_i z_j + \sum_{i=1}^{n} \alpha_i z_i + c$. Pick $N \in_R \{0,1\}^{n \times n}$ and $z \in \{0,1\}^n$ and check if $B(M+N) - B(N) + A(\alpha + z) - A(z) + c = 0$.

Thus we have a PCP with $16 = O(1)$ queries and $O(n^2)$ randomness. Furthermore, it is clear that the above test has perfect completeness. Using testability of WH-codes and Freivald's Quadratic testing analysis [Fre79] (omitted here), we can show that there exists an $\varepsilon \in (01,)$ such that if the test accepts wit probability greater than $1 - \varepsilon$, then the circuit C is satisfiable (see scribe notes of Lecture 4 in [Har07] for the soundness analysis). We thus have,

Theorem 6.3.1. *There exists $\varepsilon \in (0,1)$ such that*

$$\textit{CIRCUIT-SAT} \in \mathsf{PCP}_{1,1-\varepsilon}(O(n^2), 14).$$

References

[BCH+96] Mihir Bellare, Don Coppersmith, Johan Håstad, Marcos A. Kiwi, and Madhu Sudan. *Linearity testing in characteristic two.* IEEE Transactions on Information Theory, 42(6):1781–1795, November 1996. (Preliminary version in 36*th FOCS*, 1995). doi:10.1109/18.556674.

[BLR93] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. *Self-testing/correcting with applications to numerical problems.* J. Computer and System Sciences, 47(3):549–595, December 1993. (Preliminary Version in *22nd STOC*, 1990). doi:10.1016/0022-0000(93)90044-W.

[Fre79] Rusins Freivalds. *Fast probabilistic algorithms.* In Jirí Becvár, ed., *Proc.* 8*th Symposium of Mathematical Foundations of Computer Science*, volume 74 of *LNCS*, pages 57–69. Springer, 1979. doi:10.1007/3-540-09526-8_5.

[Har07] Prahladh Harsha. *CMSC 39600: PCPs, codes and inapproximability*, 2007. A course on PCPs at the University of Chicago (Autumn 2007).

[HW03] Johan Håstad and Avi Wigderson. *Simple analysis of graph tests for linearity and PCP.* Random Structures and Algorithms, 22(2):139–160, 2003. (Preliminary Version in *18th IEEE Conference on Computational Complexity*, 2001). doi:10.1002/rsa.10068.

[Sch80] Jacob T. Schwartz. *Fast probabilistic algorithms for verification of polynomial identities.* J. ACM, 27(4):701–717, October 1980. doi:10.1145/322217.322225.

[Sud99] Madhu Sudan. *6.893: Approximability of optimization problems*, 1999. (A course on Approximability of Optimization Problems at MIT, Fall 1999).

[SW06] Amir Shpilka and Avi Wigderson. *Derandomizing homomorphism testing in general groups.* SIAM J. Computing, 36(4):1215–1230, 2006. (Preliminary version in *36th STOC*, 2004). doi:10.1137/S009753970444658X.

[Zip79] Richard Zippel. *Probabilistic algorithms for sparse polynomials.* In Edward W. Ng, ed., *Proc. International Symposium of Symbolic and Algebraic Computation (EUROSAM)*, volume 72 of *LNCS*, pages 216–226. Springer, 1979. doi:10.1007/3-540-09519-5_73.