18.408 Topics in Theoretical Computer Science Fall 2022
Lectures 4,5,6

Dor Minzer

In this lecture we make the first step in the proof of the PCP theorem, and establish seemingly strong inapproximability result for solving quadratic equations. This construction though will not be local at all (each one will involve all of the variables), and we will turn our attention into improving upon the locality of the construction. Towards this end, we will present the sum-check protocol and low-degree extensions.

1 Quadratic Solvability

Definition 1.1 (Quadratic-Solvability over $\mathbb{F}_q$ with locality r)**.** *For a field $\mathbb{F}_q$ and $r \in \mathbb{N}$, an instance of $QS_{q,r}$ consists of a set of variables $X = \{x_1, \ldots, x_n\}$ and a set of equations $E = \{e_1, \ldots, e_m\}$, where each equation $e \in E$ is a quadratic equation in the X's involving at most r of the variables.*

Given an instance of $\text{QS}_{q,r}$, the goal is to find an assignment to the variables, namely $A\colon X \to \mathbb{F}_q$, that satisfies as many of the equations as possible.

For example, the following (X, E) is an instance of $\text{QS}_{q,r}$: take $X = \{x_1, \ldots, x_n\}$ and the equations $x_1x_2 + x_3 = 0$, $x_1^2 - 7x_4x_5 + x_2x_3 = 2$.

Abusing notations we denote by $\text{QS}_{q,r}$ the language consisting of all satisfiable instances, namely

$$\text{QS}_{q,r} = \{\, (X, E) \mid \exists A\colon X \to \mathbb{F}_q \text{ that satisfies all of the equations in } E\}\,.$$

We also consider the corresponding gap problem, $\text{QS}_{q,r}[c, s]$ for $0 < s \leqslant c \leqslant 1$, in which one is given an instance promised to be at least c satisfiable or at most s satisfiable, and the goal is to distinguish between these two cases.

We show that the Quadratic Solvability problem is NP-hard using the classical theory of NP-hardness. Then, we will use ideas from previous lectures, we show that the gap version of Quadratic Solvability is also NP-hard.

1.1 NP-hardness of Quadratic Solvability

First, we show that for some $r \in \mathbb{N}$ ($r = 5$ will be enough), the problem $\mathsf{QS}_{q,r}$ is NP-hard for all q. The idea is to start with the NP-hard 3-SAT problem, and arithmetize the clauses as equations.

Theorem 1.2. *For $r = 6$ and any field $\mathbb{F}_q$, the problem $QS_{q,r}$ is NP-hard.*

Proof. We reduce from 3-SAT. Given a 3-CNF formula $\phi = C_1 \wedge C_2 \wedge \ldots \wedge C_m$, we construct an instance (X', E') of quadratic solvability as follows. First, for every variable x_i in ϕ we construct a variable x'_i in X'. Second, for every pair of variables x_i, x_j we construct a variable $x'_{i,j}$ in X'.

We will want the variables x'_i to be Boolean valued (representing the value of x_i), and want the value of $x'_{i,j}$ to be equal to the value of x_ix_j. To implement these we add the equations $x'_i = x'^2_i$ for all i and $x'_{i,j} = x'_ix'_j$ for all i, j.

Finally, we want to write down an equation that checks whether each clause in ϕ holds. Thus, fix C to be a clause in ϕ, and assume without loss of generality that $C = (x_1 \vee x_2 \vee x_3)$. Note that this can be equivalently written as an equation as $(1 - x_1)(1 - x_2)(1 - x_3) = 0$, however this would be an equation of degree 3. Instead, we expand the left hand side, and use our $x'_{i,j}$ variables to express it as a quadratic equation: $1 - x'_1 - x'_2 - x'_3 + x'_{1,2} + x'_{2,3} + x'_{1,3} - x'_{1,2}x'_3 = 0$, and we add this equation to E'. We remark that if C has a negation in it, say $C = (\bar{x}_1 \vee x_2 \vee x_3)$, then the same idea works replacing x_1 with $1 - x_1$ (starting with the equation $x_1(1 - x_2)(1 - x_3) = 0$).

Completeness. We show that if ϕ is satisfiable, then (X', E') is satisfiable. Indeed, if $A\colon \{x_1, \ldots, x_n\} \to \{0, 1\}$ satisfies all clauses in ϕ, then we can define $B\colon X' \to \{0, 1\}$ by setting $B(x'_i) = A(x_i)$ and $B(x'_{i,j}) = A(x_i)A(x_j)$, and observe that then B satisfies all of the equations in (X', E').

Soundness. We have to show that if ϕ is unsatisfiable, then (X', E') is unsatisfiable. Equivalently, we show instead that if (X', E') is satisfiable, then ϕ is satisfiable. Towards this end, suppose that $B\colon X' \to \mathbb{F}_q$ satisfies all equations. Then in particular it satisfies that $B(x'_i)^2 = B(x'_i)$ for all i, hence $B(x'_i) \in \{0, 1\}$, and as $B(x'_{i,j}) = B(x'_i)B(x'_j)$ we get that $B(x'_{i,j}) \in \{0, 1\}$ also, hence $B\colon X' \to \{0, 1\}$. We may therefore define $A\colon \{x_1, \ldots, x_n\} \to \{0, 1\}$ by $A(x_i) = B(x'_i)$, and note that since B is Boolean and satisfies the equation associated with each clause of ϕ, it follows that A satisfies all of the clauses of ϕ. □

1.2 NP-hardness of Gap Quadratic Solvability

Next, we show that how to use Theorem 1.2 to prove that the gap version of quadratic solvability is also NP-hard. Towards this end we need the following lemma.

Lemma 1.3. *Let $m, n \in \mathbb{N}$ and let $q \geqslant 4\log(mn)^2$, $s = mnq$. One can construct, in polynomial time, a matrix $M \in \mathbb{F}_q^{s\times m}$ such that the code generated by M has relative distance at least $1 - \frac{1}{\sqrt{q}}$.*

Proof. Consider the Reed-Solomon codes $C_1 = \mathrm{RS}_{d=m,q=mn}$ and $C_2 = \mathrm{RS}_{d=\log(mn),q}$, and note that the number of codewords in C_2 is at least $(\log(mn))^{\log(mn)^2} > mn$, hence we may consider the composed code $C = C_1 \circ C_2$ which has blocklength s; we do so using an appropriate linear function mapping symbols of C_1 to codewords of C_2, so that C is a linear code. Take a matrix $M \in \mathbb{F}_q^{s\times m}$ to be a generating matrix of C. We note that the relative distance of C_1 is $1 - 1/n$, and the relative distance of C_2 is $1 - \log(mn)/q$, hence the relative distance of C is at least

$$\left(1 - \frac{1}{n}\right)\left(1 - \frac{\log mn}{q}\right) \geqslant 1 - \frac{1}{\sqrt{q}}. \qquad \square$$

With this lemma in hand, we can now deduce from Theorem 1.2 that the gap version of the Quadratic Solvability problem is also NP-hard, albeit with very poor locality parameter. The idea is that, given a system of equations as in Theorem 1.2, one can construct a new system of equations in which each equation is a linear combination of equations from the original system. One can show, for instance, that if we take sufficiently many independently chosen random linear combinations of equations, then a satisfiable system would be mapped to a satisfiable instance, and an unsatisfiable instance would be mapped to a highly unsatisfiable instance. We will see that a pre-determined set of linear combinations that are dictated by the rows of the matrix M constructed in Lemma 1.3 also does the job.

Theorem 1.4. *For $q \geqslant 4\log(mn)^2$, the problem gap-QS$_{q,n}[1, \frac{1}{\sqrt{q}}]$ is NP-hard. Here, n stands for the number of variables in the system and m stands for the number of equations in the system.*

Proof. We show a polynomial time reduction from $\text{QS}_{q,6}$ to gap-$\text{QS}_{q,r=n}[1, \frac{1}{\sqrt{q}}]$, which by Theorem 1.2 implies that gap-$\text{QS}_{q,n}[1, \frac{1}{\sqrt{q}}]$ is NP-hard. Given an instance (X, E) of $\text{QS}_{q,5}$, we construct in polynomial time an instance (X', E') of quadratic solvability such that

1. If (X, E) is satisfiable, then (X', E') is satisfiable.
2. If (X, E) is unsatisfiable, the (X', E') is at most $\frac{1}{\sqrt{q}}$-satisfiable.

Towards this end, construct a matrix $M \in \mathbb{F}_q^{s \times m}$ as in Lemma 1.3 where $s = mnq$. We define (X', E') by taking $X' = X$, and associating with each row of M an equation. In words, we think of (X, E) as a system of equations and then take linear combinations of them as indicated by the matrix M. More precisely, writing $E = \{e_1, \ldots, e_m\}$ and thinking of the equation e_j as $f_j(x) = b_j$, the ith equation in (X', E'), denoted by e'_i, is given by

$$\sum_{j=1}^{m} M_{i,j} f_j(x) = \sum_{j} M_{i,j} b_j.$$

Completeness. Suppose that (X, E) is satisfiable, and let $A \colon X \to \mathbb{F}_q$ be a satisfying assignment. Then A satisfies any equation which is the result of linear combination of equations in (X, E), and hence satisfies (X', E').

Soundness. Suppose (X, E) is unsatisfiable, and let $A \colon X' \to \mathbb{F}_q$ be some assignment. Then A does not satisfy all of the equations in (X, E), hence defining the vector $v \in \mathbb{F}_q^m$ by $v_j = f_j(A) - b_j$ gives us that v is not the all 0 vector. Hence, Mv is a codeword in the code generated by M which is not identically 0, so by the distance of the code generated by M it follows that $(Mv)_i \neq 0$ on all but $\frac{1}{\sqrt{q}}$ fraction of $i = 1, \ldots, s$. In other words, for all but $\frac{1}{\sqrt{q}}$ of i's we have that $\sum_{j=1}^{m} M_{i,j}(f_j(x) - b_j) \neq 0$, implying that A satisfies at most $\frac{1}{\sqrt{q}}$ of the equations in (X', E'). □

In the proof of Theorem 1.4 we managed to get a gap between the completeness and the soundness case (and quite a large one), however as is clear from the proof, both the alphabet size (the field size) and the number of queries (i.e. the number of variables each equation depends on) are large, and we will want to reduce them.

It is possible to reduce both the locality and the alphabet size to be $O(1)$ straightaway from Theorem 1.4, however this will result in a non polynomial size PCP (in fact exponential), and we will discuss this point later on in the course. Our proof of the PCP theorem will eventually use this idea, however in order to keep the reduction to be poly-time, it is important to first sufficiently reduce the number of queries and alphabet (just as we did in error correcting codes), say to be $\text{poly}(\log \log n)$, and only then use such ideas.

2 Query Reduction via the Sum-check Protocol and Low-degree Extensions

We now turn our attention into developing the sum-check protocol, which is the key primitive that facilitates query reduction in algebraic PCP constructions.

Let $q = \text{poly}(\log(mn))$, and consider the natural verifier in the setting of gap-$\text{QS}_{q,n}[1, 1/\sqrt{q}]$. Therein, the verifier has oracle access to an assignment $A \colon \{x_1, \ldots, x_n\} \to \mathbb{F}_q$, and his task is to verify that many of the equations in the given instance (X, E) are satisfied. For that, the verifier can pick an equation $e \in E$

randomly and check whether A satisfies it or not. The issue with this approach is that the verifier has to read the entire table of values of A to execute this plan, since each equation $e \in E$ depends on all of the variables of the system. The question is, therefore, how can the verifier check whether an equation of the form

$$\sum_{i=1,j=1}^{n} a_{i,j} x_i x_j + \sum_{i=1}^{n} b_i x_i = c$$

holds, while making less queries to A? This is clearly impossible if the verifier is only given access to A, but it turns out to be possible if the verifier is supplied with additional information!

2.1 The Low-degree Extension

Towards this end, let $\mathbb{H} \subseteq \mathbb{F}_q$ be a set whose size is to be determined shortly, and let $m \in \mathbb{N}$ be an integer so that $|\mathbb{H}|^m = n$. For this, it suffices to take $d = |\mathbb{H}| = \log(n)$ and $m = \frac{\log n}{\log \log n}$. Thus, we can identify the set of variables $[n]$ with the cube $\mathbb{H}^m$, and re-indexing them accordingly the above equation becomes

$$\sum_{\vec{i},\vec{j} \in \mathbb{H}^m} a_{\vec{i},\vec{j}} x_{\vec{i}} \cdot x_{\vec{j}} + \sum_{\vec{i} \in \mathbb{H}^m} b_{\vec{i}} x_{\vec{i}} = c. \tag{1}$$

Thinking of the assignment A now in these notations, we have that $A \colon \mathbb{H}^m \to \mathbb{F}_q$. It is clear that A may be extended to the entire domain $\mathbb{F}_q^m$ in many ways, however there is one extension, the so-called low-degree extension, which will be of utmost important to us. Roughly speaking, this is the extension of A that as a polynomial has as small as possible individual degrees. To present and prove some of its basic properties, we first introduce two basic facts about low degree polynomials that will be used many times in this course.

We begin with the classical Schwarz-Zippel lemma.

Lemma 2.1. *Suppose that $f \colon \mathbb{F}_q^m \to \mathbb{F}_q$ is a non-identically 0 polynomial of total degree d, and let $S \subseteq \mathbb{F}_q$. Then*

$$\Pr_{x \in S^m} [f(x) = 0] \leqslant \frac{d}{|S|}.$$

Proof. See problem set 1. □

We will also need a version of this lemma for individual degrees, as follows.

Lemma 2.2. *Suppose that $f \colon \mathbb{F}_q^m \to \mathbb{F}_q$ is a non-identically 0 polynomial of individual degrees at most r and $|\mathbb{H}| \geqslant r + 1$. Then there is $x \in \mathbb{H}^m$ such that $f(x) \neq 0$.*

Proof. The proof is by induction on m. For $m = 1$, this follows from the fundamental theorem of algebra. Assume $m > 1$, and write

$$f(z_1, \ldots, z_m) = \sum_{j=0}^{r} z_m^j f_j(z_1, \ldots, z_{m-1}),$$

where for each $j = 0, \ldots, r$, the function $f_j \to \mathbb{F}_q^{m-1} \to \mathbb{F}_q$ is a polynomial of individual degrees at most r. Since f is not-identically 0, there is j such that $f_j \not\equiv 0$, and by the induction hypothesis we may find a setting $(x_1, \ldots, x_{m-1}) \in \mathbb{H}^{m-1}$ such that $f_j(x_1, \ldots, x_{m-1}) \neq 0$. Thus, fixing these values the polynomial $f(x_1, \ldots, x_{m-1}, z_m)$ is a univariate polynomial of degree at most r in z_m, hence by the base case we may find a setting $z_m = x_m \in \mathbb{H}$ on which it doesn't vanish, and the proof is concluded. □

We can now prove the existence and uniqueness of the low degree extension of an assignment.

Claim 2.3. *For any $A\colon \mathbb{H}^m \to \mathbb{F}_q$, there is a unique $A_{\mathsf{extension}}\colon \mathbb{F}_q^m \to \mathbb{F}_q$ such that*

1. *$A(z) = A_{\mathsf{extension}}(z)$ for all $z \in \mathbb{H}^m$.*
2. *$A_{\mathsf{extension}}$ is a polynomial whose individual degrees are all at most $|\mathbb{H}| - 1$.*

Proof. The construction of $A_{\mathsf{extension}}$ is by interpolation. For each $z \in \mathbb{H}^m$, we can define a function $\ell_z\colon \mathbb{F}_q^m \to \mathbb{F}_q$ of individual degrees at most $|\mathbb{H}| - 1$ such that $\mathsf{supp}(\ell_z) \cap \mathbb{H}^m = \{z\}$. Indeed, one just takes

$$\ell_z(x) = \prod_{i=1}^{m} \prod_{a \in \mathbb{H} \setminus \{z_i\}} \frac{x_i - a}{z_i - a},$$

and note that $\mathsf{supp}(\ell_z) \cap \mathbb{H}^m = \{z\}$, that $\ell_z(z) = 1$ and that the individual degrees of ℓ_z are all $|\mathbb{H}| - 1$. We can thus define

$$A_{\mathsf{extension}}(x) = \sum_{z \in \mathbb{H}^m} A(z)\ell_z(x),$$

and note that $A_{\mathsf{extension}}$ has individual degrees at most $|\mathbb{H}| - 1$ and $A_{\mathsf{extension}}(z) = A(z)\ell_z(z) = A(z)$ for $z \in \mathbb{H}^m$. This proves the existence part of the claim.

For the uniqueness, suppose A' and A'' are two distinct functions satisfying the claim, and consider $B = A' - A''$. Then B has individual degrees at most $|\mathbb{H}| - 1$, and B vanishes on $\mathbb{H}^m$, and by Lemma 2.2 it follows that $B \equiv 0$. □

The function $A_{\mathsf{extension}}$ is often referred to as the low-degree extension of A and it plays a crucial role in the sum-check protocol. Instead of giving the verifier only access to the assignment A, we shall give him access to $A_{\mathsf{extension}}$ in the hope that this will help us in cutting down on the number of queries. Let us remark first that, formally speaking, the verifier is only given oracle access to some assignment $B\colon \mathbb{F}_q^m \to \mathbb{F}_q$ which is supposed to be the low-degree extension of A. Thus, the verifier will also need to make sure, somehow, that B is indeed a low-degree function; this is where the low-degree testing problem enters the picture. We ignore this issue for now, assuming the verifier is able to ensure that B is a low-degree function, and show how the protocol proceeds then. In upcoming lectures, after developing the low-degree testing machinery, we will remove this assumption.

2.2 The Sum-Check Protocol

We are now going to make use of our low-degree extension $A_{\mathsf{extension}}$ to verify that the equation

$$\sum_{\vec{i},\vec{j} \in \mathbb{H}^m} a_{i,j} x_{\vec{i}} \cdot x_{\vec{j}} + \sum_{\vec{i} \in \mathbb{H}^m} b_i x_{\vec{i}} = c \tag{2}$$

holds using much fewer queries than n. Towards this end, define the intermediate functions $f_s\colon \mathbb{H}^{2s} \to \mathbb{F}_q$ for $s = 0, 1 \ldots, m$ by[1]

$$f_s(i_1, \ldots, i_s, j_1, \ldots, j_s) = \sum_{\substack{\vec{\alpha} \in \mathbb{H}^m, \alpha_\ell = i_\ell, \\ \vec{\beta} \in \mathbb{H}^m, \beta_\ell = j_\ell \\ \text{for } \ell = 1, \ldots, s}} a_{\vec{\alpha},\vec{\beta}} A(\vec{\alpha}) \cdot A(\vec{\beta}) + \frac{1}{|\mathbb{H}|^s} \sum_{\vec{\alpha} \in \mathbb{H}^m, \alpha_\ell = i_\ell \text{ for } \ell = 1, \ldots, s} b_{\vec{\alpha}} A(\vec{\alpha}). \tag{3}$$

[1] By $\frac{1}{|\mathbb{H}|^s}$, we mean the inverse of $|\mathbb{H}|^s \in \mathbb{F}_q$.

In words, the function f_s represents partial sums similar to the ones in (2) in which a prefix of the indices $\vec{i}$ and $\vec{j}$ has been fixed to be according to the input of f_s.

Note that in this language, the equation that we want to verify is that $f_0 = c$. Additionally, note that we have, for all s, that

$$f_s(i_1, \ldots, i_s, j_1, \ldots, j_s) = \sum_{i_{s+1}, j_{s+1} \in \mathbb{H}} f_{s+1}(i_1, \ldots, i_s, i_{s+1}, j_1, \ldots, j_s, j_{s+1}), \tag{4}$$

and that

$$f_m(i_1, \ldots, i_m, j_1, \ldots, j_m) = a_{i_1,\ldots,i_m,j_1,\ldots,j_m} A(i_1, \ldots, i_m) \cdot A(j_1, \ldots, j_m) + b_{1_1,\ldots,1_m} A(i_1, \ldots, i_m). \tag{5}$$

Finally, note that the function f_m is composed only of $O(1)$ entries of A (to be exact, 2).

This suggests a recursive approach: to verify that $f_0 = c$, reduce that to verifying $f_1(i_1, j_1) = c_{i_1,j_1}$ for some i_1, j_1, further reduce that to $f_2(i_1, i_2, j_1, j_2) = c_{i_1,i_2,j_1,j_2}$ for some i_1, i_2, j_1, j_2, and continue until we need to verify some value of f_m, which can be done by appealing to the table of values A. To carry out this recursion though we need some redundancies, hence we consider the low-degree extensions of each f_s. Abusing notations, we will refer to the low-degree extension of f_s by the same notation, $f_s \colon \mathbb{F}_q^{2s} \to \mathbb{F}_q$, and we can now present the sum-check protocol.

The inputs to the sum-check protocol are the assignment $A_0 \colon \mathbb{F}_q^m \to \mathbb{F}_q$ which has individual degrees at most $|\mathbb{H}| - 1$, as well as functions $g_{s,\vec{i}',\vec{j}'} \colon \mathbb{F}_q^2 \to \mathbb{F}_q$ for each $s = 1 \ldots, m$ and $\vec{i}', \vec{j}' \in \mathbb{F}_q^{s-1}$ of individual degrees at most $|\mathbb{H}| - 1$. The goal is to verify that A_0 satisfies (1), and the intention is that $g_{s,\vec{i}',\vec{j}'}$ is the restriction of function f_s where the first $s - 1$ coordinates of i and of j are set according to $\vec{i}'$ and $\vec{j}'$. We proceed as follows:

1. Verify that $g_0 = c$, else reject.
2. Verify that $\sum_{h,h' \in \mathbb{H}} g_1(h, h') = g_0$, else reject.
3. Set $s = 1$.
4. While $s \leqslant m$:
   (a) Choose $i_s, j_s \in \mathbb{F}_q$ randomly, let $\vec{i}'_s = (i_1, \ldots, i_s)$ and $\vec{j}'_s = (j_1, \ldots, j_s)$.
   (b) Verify that $\sum_{h,h' \in \mathbb{H}} g_{s+1,\vec{i}'_s,\vec{j}'_s}(h, h') = g_{s,\vec{i}'_{s-1},\vec{j}'_{s-1}}(i_s, j_s)$, else reject.
   (c) Increase s by 1.
5. Verify that[2]

   $$g_{m,\vec{i}'_{m-1},\vec{j}'_{m-1}}(i_m, j_m) = a_{i_1,\ldots,i_m,j_1,\ldots,j_m} A_0(i_1, \ldots, i_m) \cdot A_0(j_1, \ldots, j_m) + b_{i_1,\ldots,i_m} A_0(i_1, \ldots, i_m),$$

   else reject.

Below we prove the correctness of the Sum-check Protocol.

[2]We remark that here, the coefficients $a_{i_1,\ldots,i_m,j_1,\ldots,j_m}$ are the low-degree extension of the original coefficients $a_{i_1,\ldots,i_m,j_1,\ldots,j_m}$ for $i_1, \ldots, i_m, j_1, \ldots, j_m \in \mathbb{H}$. In contrast to the functions f_s and $A_{\mathsf{extension}}$, the verifier is aware of all of the values of these coefficients, hence he can compute the low degree extension.

Lemma 2.4. *Suppose that $A_0 \colon \mathbb{F}_q^m \to \mathbb{F}_q$ is a function with individual degrees at most $|\mathbb{H}| - 1$, and $g_{s,\vec{i}',\vec{j}'} \colon \mathbb{F}_q^2 \to \mathbb{F}_q$ for $s = 0, \ldots, m$, $\vec{i}', \vec{j}' \in \mathbb{F}_q^{s-1}$ are functions of individual degrees at most $|\mathbb{H}| - 1$. Then*

1. ***Completeness:*** *If A_0 is the low degree extension of an assignment satisfying* (1)*, and the functions $g_{s,\vec{i}',\vec{j}'}$ are equal to the appropriate restrictions of the functions f_s defined above for A_0, then the sum-check protocol accepts with probability* 1.
2. ***Soundness:*** *if A_0 doesn't satisfy* (1)*, then the sum-check protocol accepts with probability at most $\frac{2dm}{q}$.*

Proof. We begin by proving the completeness of the protocol.

Completeness. It is clear that the protocol passes the first checks, and we analyze the checks in the fourth and fifth steps. We focus on the fourth item as the arguments are the same. For notational simplicity, we do the proof for $s = 1$; the argument is identical for $s > 1$. By (4), we have that

$$f_1(i,j) = \sum_{h,h' \in \mathbb{H}} f_2(i,h,j,h')$$

for all $i, j \in \mathbb{H}$. From the uniqueness of the low-degree extension of both sides, as function of i, j, it follows that the low degree extension of the left hand side is equal to the low degree extension of the right hand side. However, since taking low-degree extension is a linear operator (i.e., the low degree extension of $f + f'$ is the sum of the low degree extension of f and the low degree extension of f'), it follows that

$$g_1(i,j) = f_1(i,j) = \sum_{h,h' \in \mathbb{H}} f_2(i,h,j,h') = \sum_{h,h' \in \mathbb{H}} g_{2,i,j}(h,h')$$

for all $i, j \in \mathbb{F}_q$.

Soundness. For A_0, we denote by f_s the functions as defined by (3) for A_0, and abusing notation we also denote their low degree extension by f_s. We begin with an informal explanation of the argument. If the sum check protocol accepts, then the check that $g_0 = c$ passes, but as A_0 does not solve (1) it follows that $f_0 \neq c$, hence $g_0 \neq f_0$. Then, the protocol checks that the sum of values of g_1 is g_0, and by definition the sum of values of f_1 is f_0; thus, as $g_0 \neq f_0$, it follows that $g_1 \neq f_1$, and hence choosing random $i, j \in \mathbb{F}_q$ gives with high probability that $g_1(i,j) \neq f_1(i,j)$. Repeating this argument, we get that with high probability the value of g_m we look at does not coincide with the value of f_m, however the last check in the process verifies exactly that, hence the protocol would reject.

Formally, let E_s be the event that

$$g_{s,\vec{i}'_{s-1},\vec{j}'_{s-1}}(i_s, j_s) = f_s(\vec{i}'_{s-1}, i_s, \vec{j}'_{s-1}, j_s),$$

and let E be the event that the sum-check protocol accepts. First, note that $E \subseteq E_m$; indeed, if the sum-check protocol accepts, the last check passing is equivalent to the fact that E_m holds. It follows that

$$\Pr[E] = \Pr\left[\bigcup_{s=1}^{m} E_s \cap E\right] \leqslant \sum_{s=1}^{m} \Pr\left[\bar{E}_{s-1} \cap E_s \cap E\right] \leqslant \sum_{s=1}^{m} \Pr\left[E_s \cap E \mid \bar{E}_{s-1}\right].$$

If E_{s-1} fails, then $g_{s-1,\vec{i}'_{s-2},\vec{j}'_{s-2}}(i_{s-1},j_{s-1}) \neq f_{s-1}(\vec{i}'_{s-2},i_{s-1},\vec{j}'_{s-2},j_{s-1})$, so for the check of the sum-check protocol to pass in iteration s, it must be that the functions $g_{s,\vec{i}'_{s-1},\vec{j}'_{s-1}}(\star,\star)$ and $f_s(\vec{i}'_{s-1},\star,\vec{j}'_{s-1},\star)$ are different (else, the sum of $g_{s,\vec{i}'_{s-1},\vec{j}'_{s-1}}(h,h')$ over $h,h' \in \mathbb{H}$ would be $f_{s-1}(\vec{i}'_{s-2},i_{s-1},\vec{j}'_{s-2},j_{s-1})$ as opposed to $g_{s-1,\vec{i}'_{s-2},\vec{j}'_{s-2}}(i_{s-1},j_{s-1})$). Thus, these are distinct univariate polynomials of degree at most $2(|\mathbb{H}|-1)$, and the probability of E_s is the same as the probability that these two functions agree on randomly chosen $i_s,j_s \in \mathbb{F}_q$, which is at most $\frac{2(|\mathbb{H}|-1)}{q}$. We conclude that

$$\Pr[E] \leqslant \sum_{s=1}^{m} \Pr\left[E_s \cap E \mid \bar{E}_{s-1}\right] \leqslant \sum_{s=1}^{m} \frac{2|\mathbb{H}|}{q} = \frac{2m|\mathbb{H}|}{q} \leqslant \frac{2dm}{q}. \qquad \square$$

So how can we use Lemma 2.4 towards improving upon the locality of the equations that we check? Note that overall, the protocol makes $(|\mathbb{H}|+1)m$ calls to functions for the g-functions, and 2 queries to the assignment A_0. Thus, overall the protocol makes $O(\log(mn)^2)$ queries to the input tables; this is much better than the $\Theta(n)$ we started with!

The most pressing issue is that for Lemma 2.4 to be useful, we must guarantee that the assignment A_0 and the tables $g_{s,\vec{i}',\vec{j}'}$ are all polynomials of individual degrees at most $|\mathbb{H}|-1$. How do we do that? Well, for the tables $g_{s,\vec{i}',\vec{j}'}$ this is quite easy in fact; we can represent the function $g_{s,\vec{i}',\vec{j}'}$ simply by its coefficients. Each one of the functions $g_{s,\vec{i}',\vec{j}'}$ is only a bi-variate function, so we can represent it by its $|\mathbb{H}|^2$ coefficients, which is still a poly-logarithmic number of symbols. Hence we can force it to be low-degree just by design. The same cannot be said about A_0; if we simply represented it by its list of coefficients we would be back to square one since there are $|\mathbb{H}|^m$ of them, which is polynomially large. We therefore need to find some other way of representing a low-degree, multi-variate polynomial in a way that enables us to check that it is indeed a low-degree polynomial while reading much less information. This will be the topic of the next lecture.

2.3 Linearizing the Sum-check Protocol

To finish the discussion about the sum check protocol, it will be convenient for us to transform the check made by the sum-check protocol into a single quadratic equation (as opposed to as it currently is as an AND of several equations), and to do so we proceed as follows.

Fix the equation e that we run the sum-check protocol, and let G_e be the table of coefficients for all the g-functions encountered throughout the protocol. We note that the randomness of the sum-check protocol is $\vec{i} = (i_1,\ldots,i_m) \in \mathbb{F}_q^m$ and $\vec{j} = (j_1,\ldots,j_m) \in \mathbb{F}_q^m$, and fixing the randomness of it, the protocol checks an AND of m linear equations over G_e, as well as a quadratic equation involving several entries from G_e and two entries from A_0. We denote as $e_{\vec{i},\vec{j},1}\ \ldots, e_{\vec{i},\vec{j},m+1}$, and we think of them as $h_{\vec{i},\vec{j},\ell}(G_e,A_0(\vec{i}),A_0(\vec{j})) = 0$ for $\ell = 1,\ldots,m+1$. We know that each equation $e_{\vec{i},\vec{j},1}$ involves at most $\mathsf{poly}(\log n)$ entries from G_e, and $e_{\vec{i},\vec{j},m+1}$ at most $\mathsf{poly}(\log n)$ entries from G_e and 2 entries of A_0.

In this language, we have proved that assuming A_0 is a low-degree function that satisfies e, with probability 1 all of the equations $e_{\vec{i},\vec{j},\ell}$ are satisfied; else with probability at most $\frac{md}{q}$ all of the equations are satisfied. Instead of checking if all of the equations $e_{\vec{i},\vec{j},\ell}$, we can take a random linear combination of them and check it. Namely, for each $\vec{v} \in \mathbb{F}_q^{m+1}$, we consider $h_{\vec{i},\vec{j},\vec{v}} = \sum_{\ell=1}^{m+1} v_\ell h_{\vec{i},\vec{j},\ell}$, and note that:

1. If G_e, A_0 are such that all of $e_{\vec{i},\vec{j},\ell}$ are satisfied, then $h_{\vec{i},\vec{j},\vec{v}}(G_e,A_0(\vec{i}),A_0(\vec{j})) = 0$.

2. Else, at least one of $e_{\vec{i},\vec{j},\ell}$ is not 0. In that case, the vector $\vec{h} = (h_{\vec{i},\vec{j},\ell}(G_e, A_0(\vec{i}), A_0(\vec{j})))_{\ell=1,\ldots,m+1}$ is not the all 0 vector, and hence $h_{\vec{i},\vec{j},\vec{v}}(G_e, A_0(\vec{i}), A_0(\vec{j})) = \left\langle \vec{h}, \vec{v} \right\rangle \neq 0$ with probability $1 - \frac{1}{q}$.

We thus present the linearized sum-check protocol. Given an equation e and inputs A_0, G_e as to the sum-check protocol, run the sum-check protocol to generate $h_{\vec{i},\vec{j},\ell}$ as above, sample $\vec{v} \in \mathbb{F}_q^m$ uniformly, and check that $\left\langle \vec{v}, \vec{h} \right\rangle = 0$ for $\vec{h} = h_{\vec{i},\vec{j},\ell}(G_e, A_0(\vec{i}), A_0(\vec{j}))$.

Note that the number of v's is $q^m = \mathsf{poly}(n, m)$, hence for each equation e in the original system this protocol generates polynomially many equations, so overall the number of equations in the new system is polynomial in n and m. Also, we have the following properties:

Lemma 2.5. *Suppose that $A_0 \colon \mathbb{F}_q^m \to \mathbb{F}_q$ is a function with individual degrees at most $d-1$, and $g_{s,\vec{i}',\vec{j}'} \colon \mathbb{F}_q^2 \to \mathbb{F}_q$ for $s = 0, \ldots, m$, $\vec{i}', \vec{j}' \in \mathbb{F}_q^{s-1}$ are functions of individual degrees at most $|\mathbb{H}| - 1$. Then*

1. ***Completeness:*** *If A_0 is the low degree extension of an assignment satisfying* (1)*, and the functions $g_{s,\vec{i}',\vec{j}'}$ are equal to the appropriate restrictions of the functions f_s defined above for A_0, then the linearized sum-check protocol accepts with probability* 1.

2. ***Soundness:*** *if A_0 doesn't satisfy* (1)*, then the linearized sum-check protocol accepts with probability at most $\frac{2dm+1}{q}$.*

18.408 Topics in Theoretical Computer Science: Probabilistically Checkable Proofs
Fall 2022