Boolean function analysis has become an indispensable tool in understanding the limits of approximation algorithms for NP-optimization problems. These are problems where good solutions may be hard to find, but once a solution is available its quality can be easily ascertained.

An important class of NP-optimization problems are *constraint satisfaction problems*. One famous example is maximum satisfiability of 3CNF clauses, or MAX-3SAT. In this problem we are given constraints of the form

$$\begin{aligned}&x_1 \vee \overline{x}_2 \vee x_3\\&\overline{x}_1 \vee x_2 \vee \overline{x}_3\\&x_1 \vee x_2 \vee x_2\\&\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3.\end{aligned}$$

and want to find an assignment that simultaneously satisfies as many of them as possible. In this example setting x_1 to false (0) and x_2, x_3 to true (1) satisfies all four constraints.

Another example is maximum solvability of linear equations modulo 2 with three variables per equation, or MAX-3LIN. Instances of this problem look like this:

$$\begin{aligned}x_1 + x_2 + x_3 &= 1\\x_1 + x_2 + x_4 &= 0\\x_1 + x_3 + x_4 &= 1\\x_2 + x_3 + x_4 &= 1.\end{aligned}$$

Given such a system of equations, how many of them can we simultaneously satisfy? In this example you can see that we cannot satisfy all four – since the left hand sides add to zero, while the right hand sides add to one – but we can satisfy three out of the four, for example by setting $x_1 = x_2 = x_3 = x_4 = 1$.

Both MAX-3SAT and MAX-3LIN are special cases of constraint satisfaction problems over binary alphabet.

1 Constraint satisfaction problems

A *q-ary constraint satisfaction problem* (qCSP) over alphabet Σ is specified by a collection of variables $x_1, \dots, x_n$ taking values in Σ and a collection of *constraints* $\phi_1, \dots, \phi_m \colon \Sigma^q \to \{0, 1\}$, where constraint ϕ_j is associated with a sequence of q variables $x_{j_1}, \dots, x_{j_q}$. We say an assignment $\mathbf{x} = x_1 \dots x_n \in \Sigma^n$ satisfies constraint j if $\phi_j(x_{j_1}, \dots, x_{j_q}) = 1$. We say a qCSP instance is *satisfiable* if there exists an assignment that simultaneously satisfies all the constraints.

In MAX-3SAT $q = 3$, $\Sigma = \{0, 1\}$ and the constraints are disjunctions of literals. In MAX-3LIN, $q = 3$, $\Sigma = \{0, 1\}$, and the constraints are linear equations in three variables modulo 2.

In general we are often interested in the following kind of problem: Given a qCSP instance where the constraints are of a specific type, how should we go about finding an assignment that satisfies as many of them as possible? We can always try brute-force search over all possible assignments, but this takes exponential time.

Is it possible to do better? In the case of MAX-3SAT, the theory of NP-completeness tells us that if we find an optimal solution substantially faster than by brute-force search, then we could do so for every problem in NP, which is viewed as an unlikely state of things. This is true even if the MAX-3SAT instance is completely consistent: Even if there exists an assignment that satisfies all the constraints, finding such an assignmnent would take an inordinate amount of time in the worst case.

What about MAX-3LIN? If all constraints are simultaneously satsifiable, then we can find a satisfying assignment, i.e. a solution to the system of equations, by Gaussian elimination (or related linear algebra techniques). But what the equations are inconsistent? Can we still find an assignment that satisfies a large fraction of them? Notice that in expectation, a random assignment will satisfy half the equations. It turns out that this is essentially the best possible performance any efficient algorithm can guarantee in the worst case.

We will use the following terminology: We say a task is NP-hard if given an algorithm that achieves this task in time at most $t(n) \geq n$ on all instances of size n, for every NP problem there exists some other algorithm that solves all instances of size n in time $t(p(n))$ for some polynomial p. If $t(n)$ grows at a rate slower than 2^{n^ε} for every $\varepsilon > 0$ this is considered unlikely.

Theorem 1 (Håstad)**.** *For any constants $\eta, \varepsilon > 0$ the following task is* NP*-hard: Given a MAX-3LIN instance in which at least a $1 - \eta$ fraction of the constraints are simultaneously satisfiable, return an assignment that satisfies at least $(1 + \varepsilon)/2$ of them.*

The proof of this theorem consists of several steps, the last of which uses Fourier analysis. We will work out that part carefully but first let us give a rough sketch of what happens before Fourier analysis comes into play.

2 The PCP theorem and parallel repetition

The PCP theorem says that approximate optimization is hard in general, but does not give precise quantitative information about the parameters involved:

Theorem 2. *There exists an alphabet Σ and constants q and $\varepsilon > 0$ for which the following task is* NP*-hard: Given a satisfiable qCSP instance over Σ, find an assignment that satisfies at least a $1 - \varepsilon$ fraction of the constraints.*

Once we have this general form of the theorem, we can make some simplifying assumptions. Formally, we will reduce the qCSP instance Φ from the PCP theorem to a 2CSP instance Ψ which also satisfies the theorem and has some additional nice properties.

The instance Ψ two kinds of variables: In addition to the variables $x_1, \ldots, x_n$ from Ψ, it also has variables $y_1, \ldots, y_m$, each taking value in Σ^q. When x satisfies Φ, y_j is supposed to encode the restriction of $\mathbf{x}$ on coordinates $(j_1, \ldots, j_q)$.

The constraints of Ψ will encode two requirements: (1) that y_j satisfies ϕ_j and (2) that $\mathbf{y}$ is consistent with $\mathbf{x}$ (i.e. that the k-th coordinate of y_j is indeed equal to x_{j_k}). Formally, Ψ will have

qm constraints $\psi_{jk}(y_j, x_{j_k})$, $1 \le j \le m, 1 \le k \le q$ where

$$\psi_{jk}(y_j, x_{j_k}) := \phi_j(y_j) \text{ is true and the } k\text{th coordinate of } y_j \text{ equals } x_{j_k}.$$

Now suppose we have an algorithm that, given a satisfiable instance Ψ, finds an assignment satisfying a $1 - \varepsilon/q$ fraction of constraints. We will use this algorithm to do the analogous thing for Φ. So suppose Φ is satisfiable. Then by construction, so is Ψ, so we can find an assignmnent $(\mathbf{y}, \mathbf{x})$ that satisfies $1 - \varepsilon/q$ fraction of the constraints ψ_{ji}. We claim that $\mathbf{x}$ must satisfy a $1 - \varepsilon$ fraction of the constraints ϕ_j. For if $\mathbf{x}$ violates some constraint ϕ_j, then it must be that either x_{j_k} differs from the kth coordinate of y_j, in which case ψ_{ji} is violated, or if not then $\phi_j(y_j)$ must be false, so $\psi_{j1}, \dots, \psi_{jq}$ are all violated. So every constraint ϕ_j that is violated by $\mathbf{x}$ yields at least one constraint ψ_{jk} that is violated by $(\mathbf{y}, \mathbf{x})$.

This argument shows that without loss of generality, in Theorem 2 we may assume that $q = 2$ and the instance is of the "type" Ψ. Specifically, we may assume that:

1. The instance is *bipartite*: The variables come partitioned into two sets $y_1, \dots, y_m$ and $x_1, \dots, x_n$ so that the first variables in every constraint is some y_i and the second variable is some x_j,
2. The constraints are *projections*: For every constraint $\psi_{ji}(y_j, x_i)$ and every assignment to y_j, there is at most one assignment $\pi_{ji}(y_j)$ to x_i that makes $\psi_{ji}(y_j, x_i)$ true. (For convenience we relabeled the constraint $\psi_{jk}(y_j, x_{j_k})$ to $\psi_{ji}(y_j, x_i)$.)

Parallel repetition What we will need in order to understand the hardness of MAX-3LIN is following strenghtening of the PCP theorem:

Theorem 3. *For every $\gamma > 0$ there exists an alphabet Σ such that the following task is* NP*-hard: Given a satisfiable 2CSP bipartite instance with projection constraints over Σ, find an assignment that satisfies at least a γ-fraction of the constraints.*

The main difference between this statement and the original PCP theorem is that the algorithm here is merely required to satisfy a small γ-fraction of the constraints, and not a $1 - \varepsilon$ fraction of them for some small ε. One transformation that allows us to go from the original version to this stronger version is *parallel repetition*.

Given a 2CSP instance Ψ, the t-fold parallel repetition Ψ^t of Ψ is the following 2CSP. If Ψ has variables $x_1, \dots, x_n, y_1, \dots, y_m$ taking values in Σ, then Ψ^t has n^t variables $x_{i_1 \dots i_t}$, $i_1, \dots, i_t \in [n]$ and m^t variables $y_{j_1 \dots j_t}$, where $j_1, \dots, j_t \in [m]$, taking values in Σ^t. The "intended assignment" to $x_{i_1 \dots i_t}$ is $(x_{i_1}, \dots, x_{i_t})$ and similarly for the ys.

The constraints of Ψ^t are as follows: For every t-tuple of constraints $\psi_{ji_1}(y_{j_1}, x_{i_1}), \dots, \psi_{ji_t}(y_{j_t}, x_{i_t})$ of Ψ there is a constraint $\psi_{ji_1 \dots ji_t}(y_{j_1 \dots j_t}, x_{i_1 \dots i_t})$ which evaluates to true if ψ_{ji_k} evaluates to true on the kth coordinates of $y_{j_1 \dots j_t}$ and $x_{i_1 \dots i_t}$ for all k between 1 and t.

By construction, if Ψ is satisfiable so is Ψ^t (because if $(\mathbf{x}, \mathbf{y})$ is a satisfying assignment for Ψ then the intended assigment induced by $(\mathbf{x}, \mathbf{y})$ is a satisfying assignment for Ψ^t). Now suppose an algorithm managed to find an assignment $(\mathbf{y}^t, \mathbf{x}^t)$ that satisfies a γ-fraction of the constraints of Ψ^t. We would like to use this assignment to get an assignment $(\mathbf{y}, \mathbf{x})$ that satisfies a $1 - \varepsilon$ fraction of the constraints of Ψ. If $(\mathbf{y}^t, \mathbf{x}^t)$ was one of the intended assignments obtained from $(\mathbf{y}, \mathbf{x})$, we could argue like this. Suppose $(\mathbf{y}, \mathbf{x})$ violated at least an ε fraction of the constraints $\psi_{ji}(y_j, x_i)$. Then a random constraint $\psi_{ji_1 \dots ji_t}$ of Ψ^t is satisfied by $(\mathbf{y}^t, \mathbf{x}^t)$ if and only if all the constraints $\psi_{ji_1}, \dots, \psi_{ji_t}$ are satisfied simultaneously by $(\mathbf{y}, \mathbf{x})$. Since these are independent this happens with probability at most $(1-\varepsilon)^t < \gamma$ if t is a sufficiently large constant.

Unfortunately, there is no reason to assume that $(\mathbf{y}^t, \mathbf{x}^t)$ is one of the intended assignments and a much more elaborate argument is necessary to complete the proof.

We will now show how to derive Theorem 1 from Theorem 3.

3 The long code

The proof strategy will go like this. We will take the 2CSP instance Ψ from Theorem 3 and reduce it to a 3LIN instance. The reduction will look as follows. We replace each variable y_j and x_i (that takes values in some large alphabet Σ) by a collection of variables Y_j and X_i taking $\{0,1\}$-values. We will think of Y_j, X_i as strings in $\{0,1\}^n$ that "encode" the values of y_j, x_i. Then each constraint $\psi_{ji}(y_j, x_i)$ will be replaced by a collection of 3LIN constraints which check three things: (1) Y_j is a proper encoding of some y_j; (2) X_i is a proper encoding of some x_i; and (3) The string encoded by Y_j and the string encoded by X_i satisfy the projection constraint ψ_{ji}.

To see how this may be possible let us first forget about the third requirement and focus on the first two. Given a string $x \in \Sigma$, how can we come up with a boolean encoding X of x so that proper encodings are specified by 3LIN constraints? In fact we already did this: If we take X to be the Hadamard encoding of x, then the statement "X is a codeword of the Hadamard code" can be represented by the linearity constraints $X(s) + X(t) + X(s+t) = 0$ for all s, t in the domain. Our analysis of the linearity test showed that if X satisfied a $(1+\varepsilon)/2$ fraction of these constraints, then it has correlation at least ε with some codeword of the Hadamard code.

Now suppose we have a single projection constraint $\psi(y, x)$, where y and x take values in Σ. This means for every y, there is at most one value $x = \pi(y)$ which makes $\psi(y, x)$ true. We are given some encodings X of x and Y of y and we want to encode the statement $x = \pi(y)$ into a collection of 3LIN formulas. If X is the Hadamard encoding of x, then $X(s) = \ell_s(x)$, where ℓ_s is the linear function $\langle s, \cdot \rangle$. Similarly, if Y is the Hadamard encoding of y, then $Y(t) = \ell_t(y)$. How can we check that $x = \pi(y)$? Suppose we observe X at position s and Y at position t. We would expect to see the values $\ell_s(x)$ and $\ell_t(y)$. If we chose s and t so that ℓ_t and $\ell_s \circ \pi$ are the same function, then we could simply check that $X(s) = Y(t)$ and this would give us evidence that $x = \pi(y)$. Notice that the constraint $X(s) = Y(t)$, or $X(s) + Y(t) = 0$, is a 2LIN constraint.

Unfortunately, unless we are very lucky with π, $\ell_s \circ \pi$ will not be a linear function at all. This suggests that it may be helpful to extend the encoding X. The Hadamard encoding of x tells us the value of all linear functions at x, but it seems that we may also want to know the values of some nonlinear ones. But as long as we can handle some nonlinear functions, why not handle all of them?

The *(binary) long code* over message set Σ encodes a message $a \in \Sigma$ by a string dict_a in $\{1, -1\}^{2^{|\Sigma|}}$. Each position of the long code is indexed by a string $s \in \{0,1\}^\Sigma$ – which can also be viewed as (the truth-table of) a function $s \colon \Sigma \to \{0,1\}$ – and the encoding of a at position s is given by $\mathrm{dict}_a(s) = (-1)^{s_a}$. A corrupted codeword f could be any function $\{0,1\}^\Sigma \to \{1,-1\}$, and the actual codewords are the *dictator functions* dict_a.

A dictatorship test We now need a test for the long code based on 3LIN constraints. We have the linearity test as a starting point. This test always accepts all the dictator functions $f(s) = s_a$, but unfortunately it also accepts all the other linear functions $\ell_c(s) = \langle c, s\rangle$ for $|c| > 1$. Can we weed out those functions where $|c| > 1$?

We won't quite achieve this, but here is one idea about how we can distinguish between light cs and heavy cs. Let η be a small constant and choose a pair s, n from $\{0,1\}^n$ independently by according to different distributions. We choose s from the uniform distribution, while each coordinate of n is chosen from the η-biased distribution. This means each coordinate is chosen independently at random but takes value 1 with some small probability η and value 0 with probability $1 - \eta$.

Now consider the event $\ell_c(s) = \ell_c(s + n)$. The probability of this event is $\frac{1}{2}(1 + (1 - 2\eta)^{|c|})$. When $|c| = 1$ this probability is $1 - \eta$ which is close to one, but as $|c|$ becomes larger the probability approaches 1/2 at an exponential rate. So if given ℓ_c we choose a random c and accept if $\ell_c(s) = \ell_c(s + n)$, we are much more likely to accept dictators than linear functions that depend on a lot of variables.

Now we combine this idea with the linearity test: Given a function $F\colon \{0,1\}^n \to \{1, -1\}$, choose inputs s, t uniformly at random and n from the η-biased distribution and accept if $F(s)F(t)F(s + t + n) = 1$.

This test accepts all dictator functions with probability $1 - \eta$. Let's see what we can say about F if the test accepts it with probability at least $(1 + \varepsilon)/2$:

$$\begin{aligned}
\varepsilon &\geq \mathrm{E}_{s,t,n}[F(s)F(t)F(s+t+n)] \\
&= \sum_{a,b,c} \hat{F}_a \hat{F}_b \hat{F}_c \, \mathrm{E}_{s,t,n}[\chi_a(s)\chi_b(t)\chi_c(s+t+n)] \\
&= \sum_{a,b,c} \hat{F}_a \hat{F}_b \hat{F}_c \, \mathrm{E}_s[\chi_{a+c}(s)] \, \mathrm{E}_t[\chi_{b+c}(t)] \, \mathrm{E}_n[\chi_c(n)] \\
&= \sum_a \hat{F}_a^3 \, \mathrm{E}_n[\chi_a(n)] = \sum_a \hat{F}_a^3 \prod_{i\colon a_i = 1} \mathrm{E}_{n_i}[(-1)^{n_i}] = \sum_a \hat{F}_a^3 (1 - 2\eta)^{|a|}.
\end{aligned}$$

Let k be the smallest integer so that $(1 - 2\eta)^{k+1} \leq \varepsilon/2$. Then by Parseval's identity

$$\sum_{a\colon |a| > k} \hat{F}_a^3 (1 - 2\eta)^{|a|} \leq (1 - 2\eta)^k \leq \varepsilon/2$$

and so

$$\varepsilon/2 \leq \sum_{a\colon |a| \leq k} \hat{F}_a^3 (1 - 2\eta)^{|a|} \leq \max\nolimits_{a\colon |a| \leq k} \hat{F}_a$$

so F must have correlation $\varepsilon/2$ with some character χ_a with $|a| \leq k$, in particular a k-junta. While we cannot say that such an F is related in any way to a dictator, this weaker conclusion will be sufficient for what we need.

4 Hard instances of MAX-3LIN

Given a 2CSP instance Ψ as in Theorem 3, we now show how to construct a 3LIN instance Ξ as in Theorem 1. We start with a satisfiable instance Ψ, argue that in the corresponding Ξ at least a

$1-\eta$ fraction of constraints can be satisfied, apply an imaginary algorithm that finds an assignment satsifying at least $(1+\varepsilon)/2$ of the constraints, and show how to convert it into an assignment that satisfies a γ-fraction of the constraints of Ψ, where $\gamma = \Omega(\eta\varepsilon^3)$.

For each variable x_i of Ψ taking values in Σ, we introduce $2^{|\Sigma|-1}$ boolean variables X_i in Ξ. Similarly for every y_j we introduce such variables Y_j. The intended assignments to X_i and Y_j are the long code encodings of x_i and y_j, namely the truth-tables of the dictator functions dict_{x_i} and dict_{y_j} with one small modification.

For a technical reason, it will be convenient to work with a slightly less redundant encoding. The dictator functions are odd: $\mathrm{dict}_a(s) = -\mathrm{dict}_a(\overline{s})$. So it is enough to specify the encodings $X_i(s)$ only for half of the inputs s; the value at the other inputs can be interpolated from the formula $X_i(\overline{s}) = -X_i(s)$ and similarly for the Y_js. This transformation is called *folding*.

We now describe the constraints of Ξ. We will specify what a random constraint of Ξ looks like. To get the instance consisting of all the constraints, we make a list of all possible random constraints.

A random constraint of Ξ is created by the following experiment. We first choose a random constraint ψ_{ij} of Ψ. We now want a linear constraint that involves exactly three of the boolean variables among X_i, Y_j and "checks" that if we view Y_j and X_i as a possibly corrupted long code encodings, then the value encoded by Y_j projects to the value encoded by X_i according the the projection π_{ij} specified by ψ_{ij}.

To understand what this constraint should look like, suppose two of the variables involved in the constraint are $X_i(s)$ and $Y_j(t)$. What should the third one be? In the intended assignment, $X_i(s)$ and $Y_j(t)$ are the dictator functions s_{x_i} and t_{y_j}, where x_i and y_j satisfy the projection constraint $\pi_{ij}(y_i) = x_j$. Recall that we need to check three things: (1) that $X_i(s)$ looks like a dictator s_x; (2) that $Y_j(t)$ looks like a dictator t_y and (3) that $\pi_{ij}(y) = x$. To achieve this, we have the following three tests:

$$
\begin{aligned}
X_i(s)X_i(s')X_i(s+s'+n) &= 1 && \text{dictatorship test for } X_i\\
Y_j(t)Y_j(t')Y_j(t+t'+n) &= 1 && \text{dictatorship test for } Y_j\\
Y_j(s\circ\pi_{ij}) &= X_i(s) && \text{consistency test for } \pi_{ij}.
\end{aligned}
$$

Doing these tests separately is wasteful. Instead we roll all three of them into one:

$$X_i(s)Y_j(t)Y_j(s\circ\pi_{ij}+t+n) = 1$$

where s and t are chosen uniformly at random, and n is chosen from the η-biased distribution.

Suppose Ψ has a satisfying assignment $(\mathbf{x}, \mathbf{y})$. Let X_i and Y_j be the long code encodings dict_{x_i} and dict_{y_j} of the ith entry of $\mathbf{x}$ and the jth entry of $\mathbf{y}$ respectively. The probability that a random constraint of Ξ is satisfied is

$$
\begin{aligned}
&\Pr_{i,j,s,t,n}[\mathrm{dict}_{x_i}(s)\mathrm{dict}_{y_j}(t)\mathrm{dict}_{y_j}(s\circ\pi_{ij}+t+n) = 1]\\
&\qquad = \Pr[s_{x_i}t_{y_j}(s\circ\pi_{ij})_{y_j}t_{y_j}n_{y_j} = 1] = \Pr[s_{x_i}t_{y_j}s_{\pi_{ij}(y_j)}t_{y_j}n_{y_j} = 1] = \Pr[n_{y_j} = 1] = 1-2\eta.
\end{aligned}
$$

If Theorem 1 was false, we would be able to efficiently find some other assignment $X_1, \ldots, X_n$, $Y_1, \ldots, Y_m$ that satisfies a $(1+\varepsilon)/2$ fraction of constraints of Ξ. We show how to use this assignmnent to produce a new one $(\mathbf{x}, \mathbf{y})$ that satisfies a γ-fraction of the constraints of Ψ.

Since $X_1, \ldots, X_n, Y_1, \ldots, Y_m$ satisfies a $(1+\varepsilon)/2$ fraction of constraints of Ξ, we must have

$$\mathrm{E}_{i,j}[\mathrm{E}_{s,t,n}[X_i(s)Y_j(t)Y_j(s\circ\pi_{ij}+t+n)]] \geq \varepsilon$$

so by Markov's inequality, $\mathrm{E}_{s,t,n}[X_i(s)Y_j(t)Y_j(s \circ \pi_{ij} + t + n)] \geq \varepsilon/2$ for at least $\varepsilon/2$ of the pairs (i, j).

Fix such a pair and to simplify notation let $X = X_i, Y = Y_i, \pi = \pi_{ij}$. Applying Fourier expansion we get

$$\begin{aligned}
\varepsilon/2 &\leq \mathrm{E}_{s,t,n}[X(s)Y(t)Y(s \circ \pi + t + n)] \\
&= \sum_{a,b,c} \hat{X}_a \hat{Y}_b \hat{Y}_c \, \mathrm{E}_{s,t,n}[\chi_a(s)\chi_b(t)\chi_c(s \circ \pi + t + n)] \\
&= \sum_{a,b,c} \hat{X}_a \hat{Y}_b \hat{Y}_c \, \mathrm{E}_s[\chi_a(s)\chi_c(s \circ \pi)] \, \mathrm{E}_t[\chi_b(t)\chi_c(t)] \, \mathrm{E}_n[\chi_c(n)].
\end{aligned}$$

Clearly $\mathrm{E}_t[\chi_b(t)\chi_c(t)] = 1$ when $b = c$ and 0 otherwise and $\mathrm{E}_n[\chi_c(n)] = (1 - 2\eta)^{|c|}$. The term

$$\mathrm{E}_s[\chi_a(s)\chi_c(s \circ \pi)] = \mathrm{Ee}_s[\langle a, s \rangle + \langle c, s \circ \pi \rangle] = \mathrm{Ee}_s\Big[\sum_x s_x \Big(a_x + \sum_{y\colon \pi(y)=x} c_y\Big)\Big].$$

Let $\mathrm{odd}(c)_x = \sum_{y\colon \pi(y)=x} c_y$. This term vanishes unless $a = \mathrm{odd}(c)$. So we get

$$\sum_c \hat{Y}_c^2 \hat{X}_{\mathrm{odd}(c)} (1 - 2\eta)^{|c|} \geq \varepsilon/2.$$

Let k be the smallest integer so that $(1 - 2\eta)^{k+1} \leq \varepsilon/4$. Then by Parseval's identity

$$\sum_{c\colon |c|>k} \hat{Y}_c^2 \hat{X}_{\mathrm{odd}(c)} (1 - 2\eta)^{|c|} \leq \varepsilon/4$$

and so

$$\begin{aligned}
\varepsilon/4 &\leq \sum_{c\colon |c|\leq k} \hat{Y}_c^2 \hat{X}_{\mathrm{odd}(c)} (1 - 2\eta)^{|c|} \\
&\leq \sum_{c\colon |c|\leq k} \hat{Y}_c^2 |\hat{X}_{\mathrm{odd}(c)}| \\
&\leq \sqrt{\sum_{c\colon |c|\leq k} \hat{Y}_c^2} \sqrt{\sum_{c\colon |c|\leq k} \hat{Y}_c^2 \hat{X}_{\mathrm{odd}(c)}^2} \\
&\leq \sqrt{\sum_{c\colon |c|\leq k} \hat{Y}_c^2 \hat{X}_{\mathrm{odd}(c)}^2}.
\end{aligned}$$

The second-to-last line follows by the Cauchy-Schwarz inequality and the last one uses Parseval's identity.

Now consider the following probabilistic algorithm for creating an assignment to Ψ: For every variable y_j, first choose $c \in \{0, 1\}^\Sigma$ with probability $\hat{Y}_{j,c}^2$, then choose y_j uniformly at random from all y such that $c_y = 1$. Similarly, for every x_j, choose $a \in \{0, 1\}^\Sigma$ with probability $\hat{X}_{j,a}^2$, then choose x_j uniformly at random from all x such that $a_x = 1$.

(But what if we happened to choose $a = 0$ or $c = 0$ and no choice of x and y is possible? This will never happen because of the folding. Folding guarantees that exactly half of the entries of X and Y are ones, and so $\hat{X}_0^2 = \hat{Y}_0^2 = 0$.)

We now argue that in expectation, this assignment satisfies at least a γ fraction of constraints of Ψ. Fix a "good" pair (i, j) for which $\mathrm{E}_{s,t,n}[X(s)Y(t)Y(s \circ \pi + t + n)] \geq \varepsilon/2$. We will show that the constraint $\psi = \psi_{ij}$ is satisfied with probability at least $\varepsilon^2/16k$. By the calculation we just did

$$\sum_{c:\ |c| \leq k} \hat{Y}_c^2 \hat{X}_{\mathrm{odd}(c)}^2 \geq \varepsilon^2/16$$

What is the probability that ψ is satisfied, i.e that $\pi(y) = x$ when x and y are chosen as above? Suppose we happened to choose $a = \mathrm{odd}(c)$. Then for every x such that $a_x = 1$, there must exist at least one y such that $c_y = 1$ and $\pi(y) = x$ (since the number of such y is odd). So the probability of choosing a y such that $\pi(y) = x$ is at least $1/|c|$ and

$$\Pr_{a,c,x,y}[\pi(y) = x] \geq \sum_c \hat{Y}_c^2 \hat{X}_{\mathrm{odd}(c)}^2 \frac{1}{|c|} \geq \sum_{c:\ |c| \leq k} \hat{Y}_c^2 \hat{X}_{\mathrm{odd}(c)}^2 \frac{1}{k} \geq \frac{\varepsilon^2}{16k}$$

Since at least an $\varepsilon/2$ fraction of pairs (i, j) is good, in expectation the assignment will satisfy at least an $\varepsilon^3/16k$ fraction of the constraints. It is possible to make this assignment explicit but let's not worry about that.