Candidate Hard Unique Game

Dana Moshkovitz
MIT

Joint work with Subhash Khot, NYU

Full version: on my homepage and to appear in STOC’16.
2LIN(p): Given a system of linear equations over variables $x_1,...,x_n$, where each equation is of the form $x_i - x_k = b_{ik}$ (mod $p$), and there is an assignment that satisfies $1-\delta$ fraction of the equations, find an assignment that satisfies as many equations as possible.

For example: $p=2$. There is an assignment that satisfies 10 equations. How many can you satisfy?

\begin{align*}
x_1 + x_2 &= 0 & x_1 + x_5 &= 0 \\
x_2 + x_4 &= 1 & x_2 + x_3 &= 1 \\
x_3 + x_5 &= 0 & x_3 + x_6 &= 0 \\
x_4 + x_3 &= 0 & x_4 + x_7 &= 0 \\
x_5 + x_6 &= 0 & x_5 + x_4 &= 1 \\
x_6 + x_7 &= 0 & x_6 + x_1 &= 1 \\
x_7 + x_1 &= 1 & x_7 + x_2 &= 1
\end{align*}
The Unique Games Conjecture

For sufficiently large $p = p(\varepsilon, \delta)$ it is NP-hard: given $2\text{LIN}(p)$ system where $1-\delta$ fraction of equations hold, to find a solution that satisfies $\varepsilon$ fraction of equations [Khot-Kindler-Mossel-O’Donnell’04 formulation; original conjecture in Khot’02].
For sufficiently large $p=p(\epsilon, \delta)$ it is NP-hard: given $2\text{LIN}(p)$ system where $1-\delta$ fraction of equations hold, to find a solution that satisfies $\epsilon$ fraction of equations [Khot-Kindler-Mossel-O’Donnell’04 formulation; original conjecture in Khot’02].

For a large family of problems (including Max-Cut, Vertex-Cover, etc etc), “basic SDP” gives the best efficient approximation algorithm [KR’03, KKMO’04, Raghavendra’08,...].
Attacks on the UGC
Attacks on the UGC

**Weak approximation:**

- For small alphabet: $1-O(\sqrt{\delta \log p})$-approx [Khot, Charikar-Makarychev-Makarychev]
- For small $\delta=\delta(n)$: e.g., $n^{-\delta/2}$-approx [Trevisan, Gupta-Talwar, Charikar et al, Chlamtac et al]
Attacks on the UGC

**Weak approximation:**
- For small alphabet: $1 - O(\sqrt{\delta \log p})$-approx
  [Khot, Charikar-Makarychev-Makarychev]
- For small $\delta = \delta(n)$: e.g., $n^{-\delta/2}$-approx
  [Trevisan, Gupta-Talwar, Charikar et al, Chlamtac et al]

**Special inputs:** E.g., algorithms for random instances
  [Arora-Khot-Kolla-Steurer-Tulsiani-Vishnoi,...]
Attacks on the UGC

**Weak approximation:**
- For small alphabet: $1 - O(\sqrt{\delta \log p})$-approx [Khot, Charikar-Makarychev-Makarychev]
- For small $\delta = \delta(n)$: e.g., $n^{-\delta/2}$-approx [Trevisan, Gupta-Talwar, Charikar et al, Chlamtac et al]

**Long time:** Time $2^{n \text{poly}(\delta)}$ [Arora-Barak-Steurer]

**Special inputs:** E.g., algorithms for random instances [Arora-Khot-Kolla-Steurer-Tulsiani-Vishnoi,...]
Attacks on the UGC

**Weak approximation:**
- For small alphabet: $1 - 0(\sqrt{\delta \log p})$-approx [Khot, Charikar-Makarychev-Makarychev]
- For small $\delta = \delta(n)$: e.g., $n^{-\delta/2}$-approx [Trevisan, Gupta-Talwar, Charikar et al, Chlamtac et al]

**Special inputs:** E.g., algorithms for random instances [Arora-Khot-Kolla-Steurer-Tulsiani-Vishnoi,...]

**Long time:** Time $2^{n \text{poly}(\delta)}$ [Arora-Barak-Steurer]

**Candidate algorithmic method:** Lasserre (Sum of Squares) hierarchy of semidefinite programs.
- Generalizes all of the above [Barak-Raghavendra-Steurer, Guruswami-Sinop].
Work towards the UGC
Work towards the UGC

**Hardness for harder problems:**
- Satisfy many equations: $\varepsilon \geq 1-(11/8)\delta$ [..., Håstad-Huang-Manokaran-O’Donnell-Wright]
- Few equations can be satisfied: $\delta \geq 1/2$ [Feige-Reichman, O’Donnell-Wright]
Work towards the UGC

**Hardness for harder problems:**
- Satisfy many equations: $\varepsilon \geq 1 - (11/8) \delta$ [Håstad-Huang-Manokaran-O’Donnell-Wright]
- Few equations can be satisfied: $\delta \geq 1/2$ [Feige-Reichman, O’Donnell-Wright]

**Hardness for special algorithms:**
Counterexamples to basic SDP & weak hierarchies [Feige-Schechtman, Khot-Vishnoi, Khot-Saket, Raghavendra-Steurer]
Work towards the UGC

**Hardness for harder problems:**
- Satisfy many equations: $\varepsilon \geq 1-(11/8)\delta$ [..., Håstad-Huang-Manokaran-O’Donnell-Wright]
- Few equations can be satisfied: $\delta \geq 1/2$ [Feige-Reichman, O’Donnell-Wright]

**Hardness for special algorithms:**
Counterexamples to basic SDP & weak hierarchies [Feige-Schechtman, Khot-Vishnoi, Khot-Saket, Raghavendra-Steurer]

**Candidate hard unique game (this work):** how might we
(i) Rule out poly-time algorithms for unique games based on plausible complexity assumptions.
(ii) Unconditionally, rule out poly-time Lasserre based algorithms for unique games.
How would a proof of the Unique Games Conjecture look like?

<table>
<thead>
<tr>
<th>Reduction:</th>
<th>3SAT</th>
<th>2LIN(p)</th>
</tr>
</thead>
<tbody>
<tr>
<td>completeness</td>
<td>satisfiable</td>
<td>(1-δ)-satisfiable</td>
</tr>
<tr>
<td>soundness</td>
<td>unsatisfiable</td>
<td>ε-satisfiable</td>
</tr>
<tr>
<td>input size</td>
<td>N</td>
<td>M=N^c</td>
</tr>
<tr>
<td>conjectured lower bound</td>
<td>$2^{\Omega(N)}$</td>
<td>$2^{\Omega(M^{1/c})}$</td>
</tr>
</tbody>
</table>
Weak Unique Games Conjecture

<table>
<thead>
<tr>
<th>Reduction:</th>
<th>random kCSP</th>
<th>$\rightarrow$ 2LIN(2)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>completeness</th>
<th>$1-o(1)$-satisfiable</th>
<th>$1-\delta$ -satisfiable</th>
</tr>
</thead>
<tbody>
<tr>
<td>soundness</td>
<td>$o(1)$-satisfiable</td>
<td>$1-A\cdot\delta$ -satisfiable for large $A$</td>
</tr>
</tbody>
</table>

Want: In soundness case, $\forall$ assignment $\text{UNSAT}(\text{assignment}) > A\cdot\text{UNSAT}($completeness$)$
# Weak Unique Games Conjecture

<table>
<thead>
<tr>
<th>Reduction:</th>
<th>2LIN(2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>random kCSP</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Completeness</th>
<th>1-o(1)-satisfiable</th>
<th>1-δ-satisfiable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Soundness</td>
<td>o(1)-satisfiable</td>
<td>1-A·δ-satisfiable for large A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input Size</th>
<th>N</th>
<th>M = 2^{βN/c}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conjectured Lower Bound</td>
<td>2^{βN}</td>
<td>M^c</td>
</tr>
</tbody>
</table>
Candidate Hard Unique Game

- We show a reduction from random $k$CSP to $2\text{LIN}(2)$.
- We prove that in the soundness case, for certain natural assignments,
  
  $\text{UNSAT}(\text{assignment}) > S \cdot \text{UNSAT( completeness)}$.
- Generalizing to all assignments, and thereby proving the weak UGC, is tightly connected to a question on Gaussian isoperimetry.
Gaussian Isoperimetry

- **Sudakov-Tsirelson, Borell 1975**: For $\mathbb{R}^n$ with Gaussian measure, the volume-$\frac{1}{2}$ sets with least surface area are half-spaces.

  Noise-Test($f: \mathbb{R}^n \rightarrow \{-1, 1\}$ where $E[f]=0$)
  - Pick Gaussian $x, y \in \mathbb{R}^n$. Set $x_{\delta, y} = e^{-\delta}x + \sqrt{1-e^{-2\delta}}y$.
  - Accept if $f(x) = f(x_{\delta, y})$.

- **Erhard’86, Carlen-Kerce’01, Chianci et al’11, Mossel-Neeman’13-14**: Half-spaces, and only half-spaces, maximize the acceptance probability. Moreover, near max acceptance probability occurs only for near half-spaces.
Gaussian Isoperimetry

- **Sudakov-Tsirelson, Borell 1975**: For $\mathbb{R}^n$ with Gaussian measure, the volume-$(1/2)$ sets with least surface area are half-spaces.

  \[
  \text{Noise-Test}(f: \mathbb{R}^n \to \{-1,1\} \text{ where } \mathbb{E}[f]=0)
  \]
  - Pick Gaussian $x, y \in \mathbb{R}^n$. Set $x_{\delta,y} = e^{-\delta}x + \sqrt{1-e^{-2\delta}}y$.
  - Accept if $f(x) = f(x_{\delta,y})$.

- **Erhard’86, Carlen-Kerce’01, Chianci et al’11, Mossel-Neeman’13-14**: Half-spaces, and only half-spaces, maximize the acceptance probability. **Moreover**, near max acceptance probability occurs only for *near* half-spaces.

In our reduction, variables correspond to points $x \in \mathbb{R}^n$. Some of the equations are of the form $x - x_{\delta,y} = 0$. 
Gaussian Isoperimetry

• **Sudakov-Tsirelson, Borell 1975**: For $\mathbb{R}^n$ with Gaussian measure, the volume-$(1/2)$ sets with least surface area are half-spaces.

\[
\text{Noise-Test}(f: \mathbb{R}^n \rightarrow \{-1, 1\} \text{ where } E[f] = 0)
\]
- Pick Gaussian $x, y \in \mathbb{R}^n$. Set $x_{\delta, y} = e^{-\delta}x + \sqrt{(1-e^{-2\delta})}y$.
- Accept if $f(x) = f(x_{\delta, y})$.

• **Erhard'86, Carlen-Kerce'01, Chianci et al'11, Mossel-Neeman'13-14**: Half-spaces, and only half-spaces, maximize the acceptance probability. **Moreover**, near max acceptance probability occurs only for near half-spaces.

• **Mossel-Neeman'16 (motivated by our work)**: For every $f$ define its shifts as $f_{\delta, y}(x) = f(x_{\delta, y})$. Then, for some $c(\delta, \gamma) > 0$,

In our reduction, variables correspond to points $x \in \mathbb{R}^n$. Some of the equations are of the form $x - x_{\delta, y} = 0$. 
Gaussian Isoperimetry

- **Sudakov-Tsirelson, Borell 1975**: For $\mathbb{R}^n$ with Gaussian measure, the volume-$\frac{1}{2}$ sets with least surface area are half-spaces.

  Noise-Test($f: \mathbb{R}^n \to \{-1, 1\}$ where $\mathbb{E}[f]=0$)
  - Pick Gaussian $x, y \in \mathbb{R}^n$. Set $x_{\delta, y} = e^{-\delta}x + \sqrt{1-e^{-2\delta}}y$.
  - Accept if $f(x) = f(x_{\delta, y})$.

- **Erhard’86, Carlen-Kerce’01, Chianci et al’11, Mossel-Neeman’13-14**: Half-spaces, and only half-spaces, maximize the acceptance probability. *Moreover*, near max acceptance probability occurs only for near half-spaces.

- **Mossel-Neeman’16 (motivated by our work)**: For every $f$ define its shifts as $f_{\delta, y}(x) = f(x_{\delta, y})$. Then, for some $c(\delta, \gamma) > 0$,

  Noise-Test
  - Pick Gaussian $x, y \in \mathbb{R}^n$. Set $x_{\delta, y} = e^{-\delta}x + \sqrt{1-e^{-2\delta}}y$.
  - Accept if $f(x) = f(x_{\delta, y})$.

  In our reduction, variables correspond to points $x \in \mathbb{R}^n$. Some of the equations are of the form $x - x_{\delta, y} = 0$.

- A hope: A stronger statement, tailored to our construction, would suffice to prove soundness.
How to prove optimal inapproximability results?

1. **PCP Theorem**: $3\text{LIN}(2)$ is NP-hard to approximate to within some constant.

2. **Parallel repetition**: Constraints on size-$r$ sets $S$ of $3\text{LIN}(2)$ variables (typically $r=O(1)$)

3. **Long code**: Constraints on encodings of assignments to sets $S$. 
How to prove optimal inapproximability results?

1. **PCP Theorem:** $3\text{LIN}(2)$ is NP-hard to approximate to within some constant.

2. **Parallel repetition:** Constraints on size-$r$ sets $S$ of $3\text{LIN}(2)$ variables (typically $r = O(1)$).

3. **Long code:** Constraints on encodings of assignments to sets $S$.

*Long code is replaced with real code.*
How to prove optimal inapproximability results?

1. **PCP Theorem:** $3\text{LIN}(2)$ is NP-hard to approximate to within some constant.

   We take all sets of $r=N^{0.99}$ variables

2. **Parallel repetition:** Constraints on size-$r$ sets $S$ of $3\text{LIN}(2)$ variables (typically $r=O(1)$)

3. **Long code:** Constraints on **encodings** of assignments to sets $S$.

   *Long code is replaced with real code*
The Long Code and Its Unique Test

**Long code:** encode $s \in \{0,1\}^n$ by $F(f) = f(s)$ $\forall f : \{0,1\}^n \rightarrow \{0,1\}$.

**Dictator view:** when representing $f$'s as vectors in $\{0,1\}^{2n}$, the encoding takes the $s$'th coordinate from each $f$.

**Codeword test:** Given $F$ that assigns each $f$ either 0 or 1,

- $\text{UNSAT}(\text{dictator}) = \delta$.
- $\text{UNSAT}(\text{junta}) = O(\delta)$ (junta is a function of $O(1)$ dictators).
- **Thm (KKMO+MOO):** $\text{UNSAT} (\text{“non-juntas”}) = \Omega(\sqrt{\delta})$. 

The Long Code and Its Unique Test

**Long code:** encode \( s \in \{0,1 \}^n \) by \( F(f) = f(s) \ \forall \ f : \{0,1 \}^n \rightarrow \{0,1 \} \).

**Dictator view:** when representing \( f \)'s as vectors in \( \{0,1 \}^{2^n} \), the encoding takes the \( s \)'th coordinate from each \( f \).

**Codeword test:** Given \( F \) that assigns each \( f \) either 0 or 1,

\[
\{0,1 \}^{2^n}
\]

\( \text{UNSAT(dictator)} = \delta \).

\( \text{UNSAT(junta)} = O(\delta) \) (junta is a function of \( O(1) \) dictators).

**Thm (KKMO+MOO):** \( \text{UNSAT(“non-juntas”)} = \Omega(\sqrt{\delta}) \).
The Long Code and Its Unique Test

**Long code:** encode \( s \in \{0,1\}^n \) by \( F(f) = f(s) \) \( \forall \) \( f : \{0,1\}^n \rightarrow \{0,1\} \).

**Dictator view:** when representing \( f \)'s as vectors in \( \{0,1\}^{2^n} \), the encoding takes the \( s \)'th coordinate from each \( f \).

**Codeword test:** Given \( F \) that assigns each \( f \) either 0 or 1,

\[
\{0,1\}^{2^n} \rightarrow \{0,1\}^n \rightarrow \{0,1\}
\]

**UNSAT(dictator) = \( \delta \).**

**UNSAT(junta) = \( O(\delta) \)** (junta is a function of \( O(1) \) dictators).

**Thm (KKMO+MOO):** \( \text{UNSAT(“non-juntas”) = } \Omega(\sqrt{\delta}). \)
The Long Code and Its Unique Test

**Long code:** encode $s \in \{0,1\}^n$ by $F(f) = f(s)$ $\forall f : \{0,1\}^n \to \{0,1\}$.  

**Dictator view:** when representing $f$'s as vectors in $\{0,1\}^{2n}$, the encoding takes the $s$'th coordinate from each $f$.  

**Codeword test:** Given $F$ that assigns each $f$ either 0 or 1,  

For every $x \in \{0,1\}^n$, let $f'(x) = f(x)$ with probability $1-\delta$.  

**UNSAT (dictator) = $\delta$.**  
**UNSAT (junta) = $O(\delta)$** (junta is a function of $O(1)$ dictators).  
**Thm (KKMO+MOO):** $\text{UNSAT}($"non-juntas") = $\Omega(\sqrt{\delta})$. 


The Long Code and Its Unique Test

**Long code:** encode \( s \in \{0,1\}^n \) by \( F(f) = f(s) \) \( \forall f : \{0,1\}^n \to \{0,1\} \).

**Dictator view:** when representing \( f \)'s as vectors in \( \{0,1\}^{2n} \), the encoding takes the \( s \)'th coordinate from each \( f \).

**Codeword test:** Given \( F \) that assigns each \( f \) either 0 or 1,

For every \( x \in \{0,1\}^n \), let \( f'(x) = f(x) \) with probability \( 1 - \delta \).

**UNSAT (dictator) = \delta.**

**UNSAT (junta) = \( O(\delta) \) (junta is a function of \( O(1) \) dictators).**

**Thm (KKMO+MOO):** \( \text{UNSAT(“non-juntas”) = } \Omega(\sqrt{\delta}) \).
Why known techniques don’t prove the UGC

**Long code:** encode $s \in \{0,1\}^n$ by $F(f) = f(s) \ \forall \ f: \{0,1\}^n \rightarrow \{0,1\}$.

**Consistency test:** For two sets of variables, $S \cap R \neq \emptyset$, given $F_S$ encoding assignment to $S$ and $F_R$ encoding assignment to $R$, check consistency on $S \cap R$. 

![Diagram](image)
Why known techniques don’t prove the UGC

**Long code:** encode $s \in \{0,1\}^n$ by $F(f) = f(s)$ $\forall f: \{0,1\}^n \rightarrow \{0,1\}$.

**Consistency test:** For two sets of variables, $S \cap R \neq \emptyset$, given $F_S$ encoding assignment to $S$ and $F_R$ encoding assignment to $R$, check consistency on $S \cap R$.

- Encoding of $S \cap R$ is a part of encoding of $S$, $R$. 
Why known techniques don’t prove the UGC

**Long code:** encode $s \in \{0,1\}^n$ by $F(f)=f(s) \forall f:\{0,1\}^n \to \{0,1\}$.

**Consistency test:** For two sets of variables, $S \cap R \neq \emptyset$, given $F_S$ encoding assignment to $S$ and $F_R$ encoding assignment to $R$, check consistency on $S \cap R$.

- Encoding of $S \cap R$ is a part of encoding of $S$, $R$.
- But a tiny part! Might not correspond to the global encoding.
Why known techniques don’t prove the UGC

**Long code:** encode $s \in \{0,1\}^n$ by $F(f)=f(s)$ $\forall f:\{0,1\}^n \rightarrow \{0,1\}$.

**Consistency test:** For two sets of variables, $S \cap R \neq \emptyset$, given $F_S$ encoding assignment to $S$ and $F_R$ encoding assignment to $R$, check consistency on $S \cap R$.

- Encoding of $S \cap R$ is a part of encoding of $S$, $R$.
- But a tiny part! Might not correspond to the global encoding.
- Can prove 3LIN hardness via “self-correction”: $f(s)=(f+g)(s)+g(s)$. 
Why known techniques don’t prove the UGC

**Long code:** encode \( s \in \{0,1\}^n \) by \( F(f)=f(s) \ \forall \ f:\{0,1\}^n \to \{0,1\} \).

**Consistency test:** For two sets of variables, \( S \cap R \neq \emptyset \), given \( F_S \) encoding assignment to \( S \) and \( F_R \) encoding assignment to \( R \), check consistency on \( S \cap R \).

- Encoding of \( S \cap R \) is a part of encoding of \( S \), \( R \).
- But a tiny part! Might not correspond to the global encoding.
- Can prove 3LIN hardness via “self-correction”: \( f(s)=(f+g)(s)+g(s) \).

**No unique consistency test!**
Hadamard, and why it doesn’t help in proving the UGC either

**Hadamard:** encode \( s \in \{0,1\}^n \) by \( F(f)=f(s) \) \( \forall \) linear \( f: \{0,1\}^k \rightarrow \{0,1\} \), i.e., \( \forall \ a \in \{0,1\}^n \), \( F(a)=<a,s> \).

Equivalently, \( s \) encoded by the linear function \( F(x)=<x,s> \).
Hadamard, and why it doesn’t help in proving the UGC either

**Hadamard:** encode $s \in \{0,1\}^n$ by $F(f)=f(s)$ $\forall$ linear $f:\{0,1\}^k \rightarrow \{0,1\}$, i.e., $\forall$ $a \in \{0,1\}^n$, $F(a)=\langle a,s \rangle$.

Equivalently, $s$ encoded by the linear function $F(x)=\langle x,s \rangle$. 
Hadamard, and why it doesn’t help in proving the UGC either

**Hadamard:** encode \( s \in \{0,1\}^n \) by \( F(f)=f(s) \) \( \forall \) linear \( f: \{0,1\}^k \rightarrow \{0,1\} \), i.e., \( \forall a \in \{0,1\}^n, F(a)=\langle a,s \rangle \).

Equivalently, \( s \) encoded by the linear function \( F(x)=\langle x,s \rangle \).

**Consistency test:** Suppose that we only wish to test consistency between \( S \) and a random set \( R \) where \( |S \cap R|=(1-\delta)n \).
Hadamard, and why it doesn’t help in proving the UGC either

**Hadamard:** encode $s \in \{0,1\}^n$ by $F(f) = f(s)$ \(\forall\) linear $f: \{0,1\}^k \rightarrow \{0,1\}$, i.e., \(\forall a \in \{0,1\}^n, F(a) = \langle a, s \rangle\).

Equivalently, $s$ encoded by the linear function $F(x) = \langle x, s \rangle$.

**Consistency test:** Suppose that we only wish to test consistency between $S$ and a random set $R$ where $|S \cap R| = (1-\delta)n$.

- Encoding of $S \cap R$ is part of encoding of $S$, $R$. 
Hadamard, and why it doesn’t help in proving the UGC either

**Hadamard:** encode $s \in \{0,1\}^n$ by $F(f)=f(s)$ \(\forall\) linear $f: \{0,1\}^k \rightarrow \{0,1\}$, i.e., \(\forall\) $a \in \{0,1\}^n$, $F(a)=<a,s>$.

Equivalently, $s$ encoded by the linear function $F(x)=<x,s>$.

**Consistency test:** Suppose that we only wish to test consistency between $S$ and a random set $R$ where $|S \cap R|=(1-\delta)n$.
- Encoding of $S \cap R$ is part of encoding of $S$, $R$.
- Union of $S \cap R$ encodings over $R$ covers $S$ encoding.
Hadamard, and why it doesn’t help in proving the UGC either

**Hadamard:** encode $s \in \{0,1\}^n$ by $F(f) = f(s)$ $\forall$ linear $f : \{0,1\}^k \to \{0,1\}$, i.e., $\forall a \in \{0,1\}^n$, $F(a) = \langle a, s \rangle$.

Equivalently, $s$ encoded by the linear function $F(x) = \langle x, s \rangle$.

**Consistency test:** Suppose that we only wish to test consistency between $S$ and a random set $R$ where $|S \cap R| = (1-\delta)n$.
- Encoding of $S \cap R$ is part of encoding of $S$, $R$.
- Union of $S \cap R$ encodings over $R$ covers $S$ encoding.
Hadamard, and why it doesn’t help in proving the UGC either

**Hadamard:** encode \( s \in \{0,1\}^n \) by \( F(f) = f(s) \) \( \forall \) linear \( f : \{0,1\}^k \rightarrow \{0,1\} \), i.e., \( \forall \ a \in \{0,1\}^n \), \( F(a) = <a,s> \).

Equivalently, \( s \) encoded by the linear function \( F(x) = <x,s> \).

**Consistency test:** Suppose that we only wish to test consistency between \( S \) and a random set \( R \) where \( |S \cap R|=(1-\delta)n \).
- Encoding of \( S \cap R \) is part of encoding of \( S, R \).
- Union of \( S \cap R \) encodings over \( R \) covers \( S \) encoding.
Hadamard, and why it doesn’t help in proving the UGC either

**Hadamard:** encode $s \in \{0,1\}^n$ by $F(f) = f(s)$ $\forall$ linear $f : \{0,1\}^k \rightarrow \{0,1\}$, i.e., $\forall$ $a \in \{0,1\}^n$, $F(a) = \langle a, s \rangle$.

Equivalently, $s$ encoded by the linear function $F(x) = \langle x, s \rangle$.

**Consistency test:** Suppose that we only wish to test consistency between $S$ and a random set $R$ where $|S \cap R| = (1-\delta)n$.

- Encoding of $S \cap R$ is part of encoding of $S$, $R$.
- Union of $S \cap R$ encodings over $R$ covers $S$ encoding.
- Given encodings $F_S$ and $F_R$, pick uniform $a \in \{0,1\}^n$ where $a_i = 0$ whenever $i \notin S \cap R$. Check $F_S(a) = F_R(a)$. 
Hadamard, and why it doesn’t help in proving the UGC either

**Hadamard**: encode \( s \in \{0,1\}^n \) by \( F(f) = f(s) \) \( \forall \) linear \( f : \{0,1\}^k \rightarrow \{0,1\} \), i.e., \( \forall a \in \{0,1\}^n, F(a) = \langle a, s \rangle \).

Equivalently, \( s \) encoded by the linear function \( F(x) = \langle x, s \rangle \).

**Consistency Test**: Suppose that we only wish to test consistency between \( S \) and \( R \) where \( |S \cap R| = (1-\delta)n \).
- Encoding of \( S \cap R \) is part of encoding of \( S, R \).
- Union of \( S \cap R \) encodings over \( R \) covers \( S \) encoding.
- Given encodings \( F_S \) and \( F_R \), pick uniform \( a \in \{0,1\}^n \) where \( a_i = 0 \) whenever \( i \notin S \cap R \). Check \( F_S(a) = F_R(a) \).
Hadamard, and why it doesn’t help in proving the UGC either

**Hadamard:** encode $s \in \{0,1\}^n$ by $F(f) = f(s)$ $\forall$ linear $f : \{0,1\}^k \rightarrow \{0,1\}$, i.e., $\forall a \in \{0,1\}^n$, $F(a) = \langle a, s \rangle$.

Equivalently, $s$ encoded by the linear function $F(x) = \langle x, s \rangle$.

**Consistency test:** Suppose that we only wish to test consistency between $S$ and a random set $R$ where $|S \cap R| = (1-\delta)n$.
- Encoding of $S \cap R$ is part of encoding of $S$, $R$.
- Union of $S \cap R$ encodings over $R$ covers $S$ encoding.
- Given encodings $F_S$ and $F_R$, pick uniform $a \in \{0,1\}^n$ where $a_i = 0$ whenever $i \notin S \cap R$. Check $F_S(a) = F_R(a)$.

**Codeword test:** Only BLR test (for uniform $a, b$, check $F_S(a + b) = F_S(a) + F_S(b)$).
Hadamard, and why it doesn’t help in proving the UGC either

**Hadamard:** encode \( s \in \{0,1\}^n \) by \( F(f) = f(s) \) \( \forall \) linear \( f: \{0,1\}^k \to \{0,1\} \), i.e., \( \forall a \in \{0,1\}^n, F(a) = \langle a, s \rangle \).

Equivalently, \( s \) encoded by the linear function \( F(x) = \langle x, s \rangle \).

**Consistency test:** Suppose that we only wish to test consistency between \( S \) and a random set \( R \) where \( |S \cap R| = (1 - \delta)n \).
- Encoding of \( S \cap R \) is part of encoding of \( S, R \).
- Union of \( S \cap R \) encodings over \( R \) covers \( S \) encoding.
- Given encodings \( F_S \) and \( F_R \), pick uniform \( a \in \{0,1\}^n \) where \( a_i = 0 \) whenever \( i \notin S \cap R \). Check \( F_S(a) = F_R(a) \).

**Codeword test:** Only BLR test (for uniform \( a, b \), check \( F_S(a + b) = F_S(a) + F_S(b) \)).
Real Code: Unique consistency test and Unique codeword test

Real code: encode \( s \in \{1, -1\}^n \) by the periodized half-space \( h(x) = \text{intveral}(\langle s, x \rangle) \) defined over \( \mathbb{R}^n \), where \( \text{interval}: \mathbb{R} \rightarrow \{-1, 1\} \) changes signs at integers.

Consistency test: For a set of variables \( S \), for a random set \( R \) where \( |S \cap R| = (1 - \delta)n \), given encodings \( F_S \) and \( F_R \), pick gaussian \( a \in \mathbb{R}^n \) where \( |a_i| \approx \delta \) whenever \( i \notin S \cap R \). Check \( F_S(a) = F_R(a) \).

Codeword test: Pick gaussian \( x \), perturb it to get \( x' = (1 - \rho)x + \theta y, \ (1 - \rho)^2 + \theta^2 = 1 \). Check \( F_S(x) = F_S(x') \).
Overall Reduction

- Start with 3LIN over variables $V$, $|V|=N$.
- For every $S \subseteq V$, $|S|=n$, $n=N^{0.99}$, block of variables, supposedly assigned $\{\text{interval}(<s,x>)\}_x$ where $s$ is assignment to $S$. Denote the assignment $F_S: \mathbb{R}^n \to \{+1,-1\}$.
- **Folding:** $\forall S, i$, $F_S(-x)=-F_S(x)$; $F_S(x_1..1+x_i..x_n)=-F_S(x_1..x_i..x_n)$.
- **Constraint Test:** Pick random $S$. Pick random constraint $\alpha$ on $S$'s variables. Pick Gaussian $x$. Check $F_S(x+\alpha)=F_S(x)$.
- **Consistency Test:** Pick random $S$, $R$ so $|S \cap R|=(1-\delta)n$. Pick Gaussian $x \in \mathbb{R}^n$ where $|x_i|\approx \delta$ whenever $i \notin S \cap R$. Check $F_S(x)=F_R(x)$.
- **Codeword Test:** Pick random $S$. Pick Gaussian $x$, perturb it to get $x'=(1-\rho)x+\theta y$, $(1-\rho)^2+\theta^2=1$. Check $F_S(x)=F_S(x')$. 
Soundness for Real Code Juntas

• **Observation:** The probability the function $F_s(x) = \text{interval}(<s_1,x>) \cdot \text{interval}(<s_2,x>)$ fails the codeword test is only twice as much as the probability that a real code fails.

• **Definition (real code junta):** A function of $O(1)$ real code codewords.

• **Main Theorem:** In soundness case, if all $F_s$’s are real code juntas, then $\text{UNSAT}\text{(assignment)} > A \cdot \text{UNSAT}\text{(completeness)}$. 
Games with Leakage

• **Direct Product Testing:**
  – A verifier picks random \( S, R \) such that \(|S \cap R| = (1-\delta)n\).
  – Sends one player \( S \). Sends another player \( R \). Each player responds with an assignment to the variables in its set that satisfies all the constraints contained in the set.
  – The verifier checks that the two assignments agree on \( S \cap R \).

• Our consistency test emulates direct product testing, but the two players receive \( x \), which reveals information about \( S \cap R \)!
  – Our construction withstands this leakage!
  – **How?** By restriction to real code juntas, the direct product players can only use limited information about \( x \).
  – We build on parallel repetition information theoretic analysis, but crucially use the structure of leakage (information on random superset of \( S \cap R \)).
Games with Leakage

- **Direct Product Testing:**
  - A verifier picks random $S$, $R$ such that $|S \cap R| = (1-\delta)n$.
  - Sends one player $S$. Sends another player $R$. Each player responds with an assignment to the variables in its set that satisfies all the constraints contained in the set.
  - The verifier checks that the two assignments agree on $S \cap R$.

- Our consistency test emulates direct product testing, but the two players receive $x$, which reveals information about $S \cap R$!
  - Our construction withstands this leakage!
  - **How?** By restriction to real code juntas, the direct product players can only use limited information about $x$.
  - We build on parallel repetition information theoretic analysis, but crucially use the structure of leakage (information on random superset of $S \cap R$).
**Constraint Satisfaction Problems**

**kCSP Input:** $N$ variables over $\pm 1$; constraints depending on $k$ variables each. Consider Hadamard constraints as [Samorodnitsky-Trevisan].

**Goal:** find an assignment to the variables that satisfies as many constraints as possible.

**Random kCSP:** Each constraint is over a random set of variables with random signs on the variables. $\#\text{constraints} \gg \#\text{variables}$. 
- If $1-o(1)$ fraction of constraints can be satisfied (YES), algorithm accepts.
- The algorithm rejects with probability $1-o(1)$ [since with this probability only $(1+o(1))(k+1)/2^k$ fraction of constraints can be satisfied (NO)].

**Tulsiani:** Random kCSP has a Lasserre integrality gap for $\Omega(n)$ rounds.