On the Power of Interactive Proofs for Learning STOC'24

Tom Gur [1] Mohammad Mahdi Jahanara [2]
Mohammad Mahdi Khodabandeh [2] Ninad Rajgopal [1]
Bahar Salamtian [3] Igor Shinkar [2]

[1]University of Cambridge

[2]Simon Fraser University

[3]Qualcomm

August 1, 2024

Our Plan

Make Goldreich-Levin Algorithm Interactive.

Our Plan

Make Goldreich-Levin Algorithm Interactive.

- GL finds all high Fourier coefficients using poly(t, n) membership queries (t is the number of high Fourier coefficients)

Our Plan

Make Goldreich-Levin Algorithm Interactive.

- GL finds all high Fourier coefficients using poly(t, n) membership queries (t is the number of high Fourier coefficients)
- The interactive version was first discussed in [GRSY21][1]

[1] Shafi Goldwasser, Guy N. Rothblum, Jonathan Shafer, and Amir Yehudayoff. Interactive proofs for verifying machine learning.

Our Plan

Make Goldreich-Levin Algorithm Interactive.

- GL finds all high Fourier coefficients using poly(t, n) membership queries (t is the number of high Fourier coefficients)
- The interactive version was first discussed in [GRSY21][1]
- [GRSY21] use poly(t, n) random examples

[1] Shafi Goldwasser, Guy N. Rothblum, Jonathan Shafer, and Amir Yehudayoff. Interactive proofs for verifying machine learning.

Our Plan

Make Goldreich-Levin Algorithm Interactive.

- GL finds all high Fourier coefficients using poly(t, n) membership queries (t is the number of high Fourier coefficients)
- The interactive version was first discussed in [GRSY21][1]
- [GRSY21] use poly(t, n) random examples
- Our work: poly(t) random examples

[1]Shafi Goldwasser, Guy N. Rothblum, Jonathan Shafer, and Amir Yehudayoff. Interactive proofs for verifying machine learning.

The Standard Goldreich-Levin Algorithm

Goldreich-Levin Algorithm

Learning Fourier coefficients

Input Oracle access to a function $f : \{0,1\}^n \to \{0,1\}$, and $0 < \tau < 1$

Output Find all Fourier coefficients with $|\hat{f}| > \tau$.

Goldreich-Levin Algorithm

Learning Fourier coefficients

Input Oracle access to a function $f : \{0,1\}^n \rightarrow \{0,1\}$, and $0 < \tau < 1$
Output Find all Fourier coefficients with $|\hat{f}| > \tau$.

Solution overview

1. Break the domain of Fourier coefficients ($\{0,1\}^n$) to subcubes by fixing coordinates.
2. If a subcube has low Fourier weight, simply discard it. Otherwise keep splitting until we get singletons.

Goldreich-Levin Algorithm

Learning Fourier coefficients

Input Oracle access to a function $f : \{0,1\}^n \to \{0,1\}$, and $0 < \tau < 1$

Output Find all Fourier coefficients with $|\hat{f}| > \tau$.

Solution overview

1. Break the domain of Fourier coefficients ($\{0,1\}^n$) to subcubes by fixing coordinates.
2. If a subcube has low Fourier weight, simply discard it. Otherwise keep splitting until we get singletons.

Query Complexity

- We fix coordinates until we get singletons.
- We need to fix n coordinates.

$\implies$ Query complexity depends on n, and requires *membership* queries.

Goldreich-Levin Algorithm cont.

Subcube notation: $01 0 * * = \{2\}, \{2, 4\}, \{2, 5\}, \{2, 4, 5\}$.

Goldreich-Levin Algorithm cont.

Subcube notation: $010{*}{*} = \{2\}, \{2, 4\}, \{2, 5\}, \{2, 4, 5\}$.

Weight of Subcube

$$\begin{aligned}\mathbf{W}(010{*}{*}) &= \hat{f}(\ \{2\}\ )^2 + \hat{f}(\ \{2,4\}\ )^2 + \hat{f}(\ \{2,5\}\ )^2 + \hat{f}(\{2,4,5\})^2 \\ &= \hat{f}(\ 01000\ )^2 + \hat{f}(\ 01010\ )^2 + \hat{f}(\ 01001\ )^2 + \hat{f}(\ 01011\ )^2.\end{aligned}$$

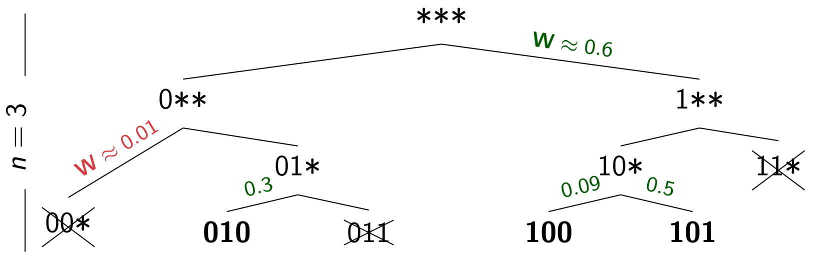
Goldreich-Levin Algorithm cont.

Subcube notation: $010{*}{*} = \{2\}, \{2,4\}, \{2,5\}, \{2,4,5\}$.

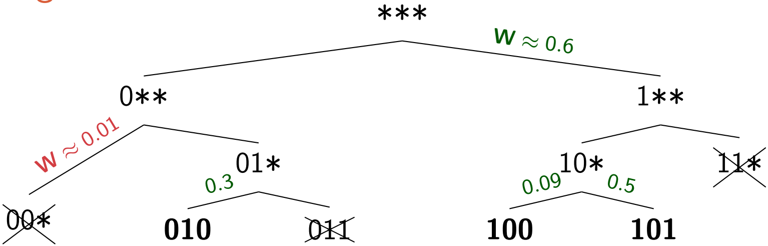
Weight of Subcube

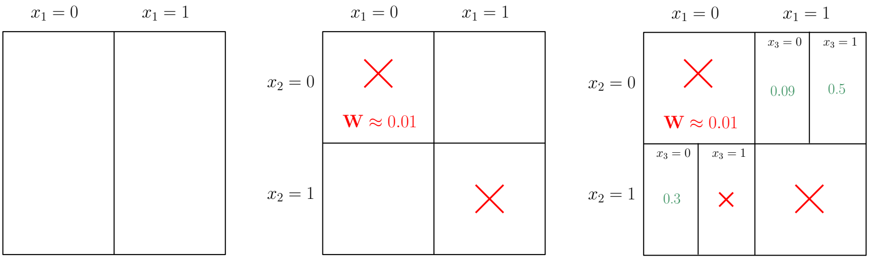
$$\mathbf{W}(010{*}{*}) = \hat{f}(\{2\})^2 + \hat{f}(\{2,4\})^2 + \hat{f}(\{2,5\})^2 + \hat{f}(\{2,4,5\})^2$$
$$= \hat{f}(01000)^2 + \hat{f}(01010)^2 + \hat{f}(01001)^2 + \hat{f}(01011)^2.$$

Input $f : \{0,1\}^3 \to \{0,1\}$, and $\tau = 0.1$



Splitting into Subcubes





Interactive Goldreich-Levin

What if we have a Prover that claims to know the top t Fourier coefficients? Can we save on query complexity?

Interactive Goldreich-Levin

What if we have a Prover that claims to know the top t Fourier coefficients? Can we save on query complexity?

Main Theorem

There is an interactive protocol for finding top t Fourier coefficients of a function $f : \{0,1\}^n \to \{0,1\}$ with error parameter ε, where the Verifier uses poly$(t, 1/\varepsilon)$ *random examples*, *independent of n*.

Interactive Goldreich-Levin

What if we have a Prover that claims to know the top t Fourier coefficients? Can we save on query complexity?

Main Theorem

There is an interactive protocol for finding top t Fourier coefficients of a function $f : \{0,1\}^n \to \{0,1\}$ with error parameter ε, where the Verifier uses $\text{poly}(t, 1/\varepsilon)$ $\underbrace{\textit{random examples}}_{\text{qualitatively efficient}}$, $\underbrace{\textit{independent of } n}_{\text{quantitatively efficient}}$.

Interactive Goldreich-Levin

What if we have a Prover that claims to know the top t Fourier coefficients? Can we save on query complexity?

Main Theorem

There is an interactive protocol for finding top t Fourier coefficients of a function $f : \{0,1\}^n \to \{0,1\}$ with error parameter ε, where the Verifier uses $\text{poly}(t, 1/\varepsilon)$ *random examples*, *independent of n*.

random examples: qualitatively efficient

independent of n: quantitatively efficient

- Our output $\Lambda = \{\gamma_1, \ldots, \gamma_t\}$ is correct with error parameter ε if for all $\gamma \in \Lambda$ and all $\beta \notin \Lambda$, it holds that $|\hat{f}(\gamma)| + \varepsilon \geq |\hat{f}(\beta)|$.

Interactive Goldreich-Levin

What if we have a Prover that claims to know the top t Fourier coefficients? Can we save on query complexity?

Main Theorem

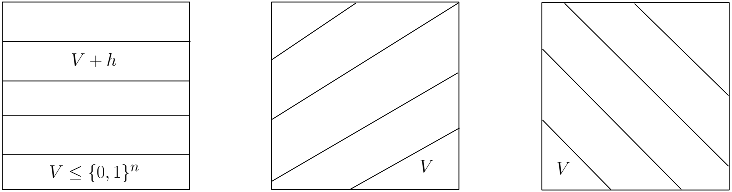
There is an interactive protocol for finding top t Fourier coefficients of a function $f : \{0,1\}^n \to \{0,1\}$ with error parameter ε, where the Verifier uses $\text{poly}(t, 1/\varepsilon)$ *random examples* (qualitatively efficient), *independent of n* (quantitatively efficient).

- Our output $\Lambda = \{\gamma_1, \ldots, \gamma_t\}$ is correct with error parameter ε if for all $\gamma \in \Lambda$ and all $\beta \notin \Lambda$, it holds that $|\hat{f}(\gamma)| + \varepsilon \geq |\hat{f}(\beta)|$.
- This means if we want $\{\gamma : |\hat{f}(\gamma)| \geq \tau\}$, then we can set $t = 4/\tau^2$, $\varepsilon = \tau/2$. Thus $\text{poly}(1/\tau)$ random examples.

Splitting the Domain (in Affine Ways)

Splitting into Affine Subspaces

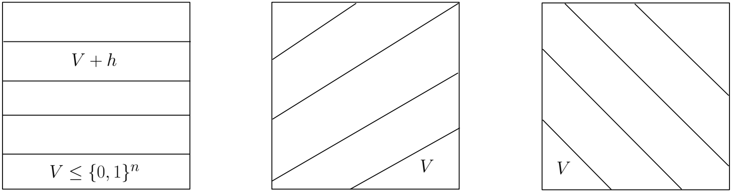
View $\{0,1\}^n$ as $\mathbb{F}_2^n$. Choose some subspace V. [GOSSW11][1]



[1] Parikshit Gopalan, Ryan O'Donnell, Rocco A Servedio, Amir Shpilka, and Karl Wimmer. Testing fourier dimensionality and sparsity.

Splitting into Affine Subspaces

View $\{0,1\}^n$ as $\mathbb{F}_2^n$. Choose some subspace V. [GOSSW11][1]

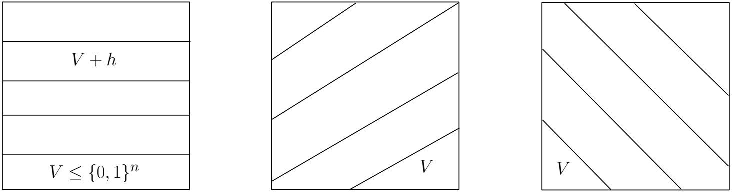


Say, we are interested in top 3 biggest Fourier coefficients.

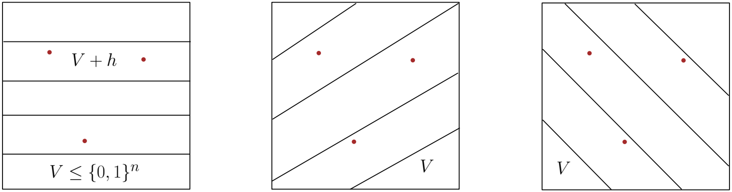
[1] Parikshit Gopalan, Ryan O'Donnell, Rocco A Servedio, Amir Shpilka, and Karl Wimmer. Testing fourier dimensionality and sparsity.

Splitting into Affine Subspaces

View $\{0,1\}^n$ as $\mathbb{F}_2^n$. Choose some subspace V. [GOSSW11][1]

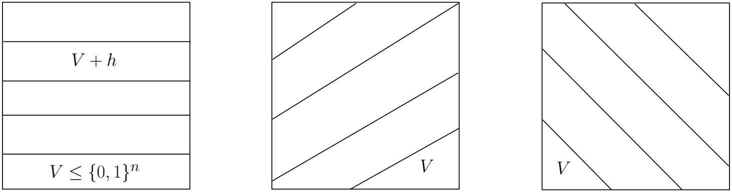


Say, we are interested in top 3 biggest Fourier coefficients.

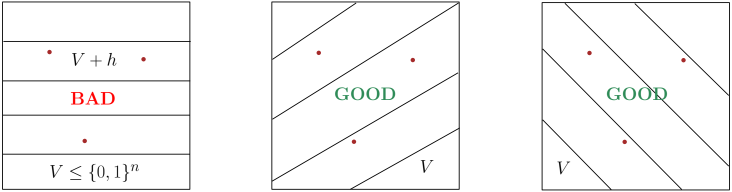


Splitting into Affine Subspaces

View $\{0,1\}^n$ as $\mathbb{F}_2^n$. Choose some subspace V. [GOSSW11][1]



Say, we are interested in top 3 biggest Fourier coefficients.



[1] Parikshit Gopalan, Ryan O'Donnell, Rocco A Servedio, Amir Shpilka, and Karl Wimmer. Testing fourier dimensionality and sparsity.

How the Splitting is Actually Done

Linear functions

Let $r, \gamma \in \{0,1\}^n$. Then $\langle r, \gamma \rangle = r_1\gamma_1 + \cdots + r_n\gamma_n$.
Think of r : random linear function; namely $\gamma \mapsto \langle r, \gamma \rangle$
Think of γ : Fourier character.

How the Splitting is Actually Done

Linear functions

Let $r, \gamma \in \{0,1\}^n$. Then $\langle r, \gamma \rangle = r_1\gamma_1 + \cdots + r_n\gamma_n$.
Think of r : random linear function; namely $\gamma \mapsto \langle r, \gamma \rangle$
Think of γ : Fourier character.

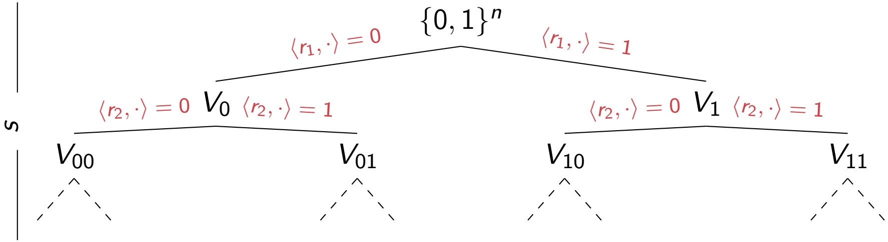
Take *linearly ind.* $r_1, r_2, \ldots, r_s \sim \{0,1\}^n$; Think of s: small.

How the Splitting is Actually Done

Linear functions

Let $r, \gamma \in \{0,1\}^n$. Then $\langle r, \gamma \rangle = r_1\gamma_1 + \cdots + r_n\gamma_n$.
Think of r : random linear function; namely $\gamma \mapsto \langle r, \gamma \rangle$
Think of γ : Fourier character.

Take *linearly ind.* $r_1, r_2, \ldots, r_s \sim \{0,1\}^n$; Think of s: small.

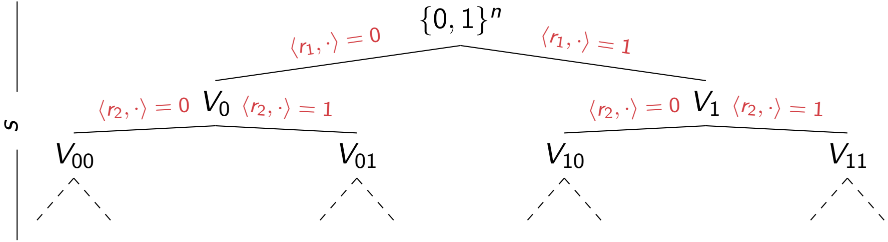


How the Splitting is Actually Done

Linear functions

Let $r, \gamma \in \{0,1\}^n$. Then $\langle r, \gamma \rangle = r_1\gamma_1 + \cdots + r_n\gamma_n$.
Think of r : random linear function; namely $\gamma \mapsto \langle r, \gamma \rangle$
Think of γ : Fourier character.

Take *linearly ind.* $r_1, r_2, \ldots, r_s \sim \{0,1\}^n$; Think of s: small.



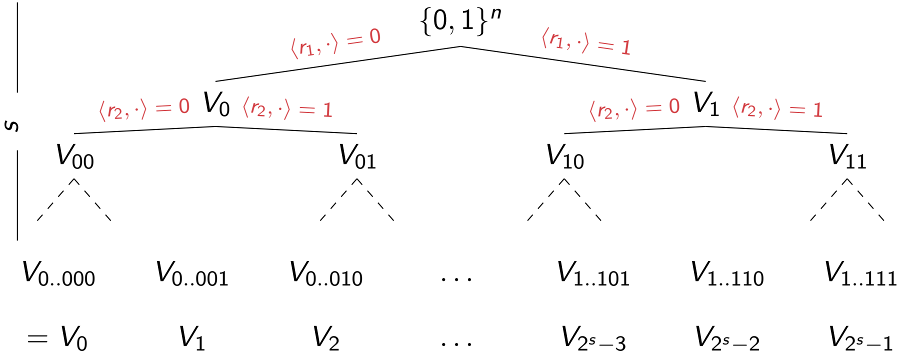
$$V_{10} = \left\{ \gamma \in \{0,1\}^n \;\middle|\; \begin{array}{l} \langle r_1, \gamma \rangle = 1 \\ \langle r_2, \gamma \rangle = 0 \end{array} \right\}$$

How the Splitting is Actually Done

Linear functions

Let $r, \gamma \in \{0,1\}^n$. Then $\langle r, \gamma \rangle = r_1\gamma_1 + \cdots + r_n\gamma_n$.
Think of r : random linear function; namely $\gamma \mapsto \langle r, \gamma \rangle$
Think of γ : Fourier character.

Take *linearly ind.* $r_1, r_2, \ldots, r_s \sim \{0,1\}^n$; Think of s: small.



Useful Observations About This Splitting

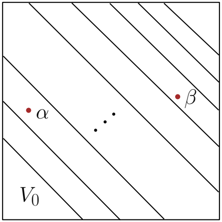
Parseval's identity

$\sum_\gamma \hat{f}(\gamma)^2 = 1.$

Useful Observations About This Splitting

Observation 1

Any two coefficients $\alpha \neq \beta \in \{0,1\}^n$ will be in different buckets **with high probability.**



Parseval's identity

$\sum_\gamma \hat{f}(\gamma)^2 = 1.$

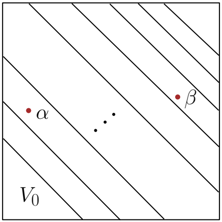
Useful Observations About This Splitting

Observation 1

Any two coefficients $\alpha \neq \beta \in \{0,1\}^n$ will be in different buckets **with high probability.**

Observation 2 - Union bound

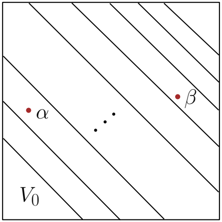
We need **small** number of random linear functions $r_1, \ldots, r_s$ to separate **all of**, say, top 3 Fourier coefficients.



Parseval's identity

$\sum_\gamma \hat{f}(\gamma)^2 = 1.$

Useful Observations About This Splitting



Observation 1

Any two coefficients $\alpha \neq \beta \in \{0,1\}^n$ will be in different buckets **with high probability.**

Observation 2 - Union bound

We need **small** number of random linear functions $r_1, \ldots, r_s$ to separate **all of**, say, top 3 Fourier coefficients.

Parseval's identity

$\sum_\gamma \hat{f}(\gamma)^2 = 1.$

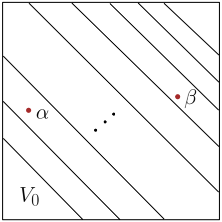
Observation 3

1. $\sum_{\gamma \in V+h} \hat{f}(\gamma)^4 = \mathbf{E}_{\substack{x,y,z \sim \{0,1\}^n \\ w \sim V^\perp}}[\chi_h(w) \cdot f(x) \cdot f(y) \cdot f(z) \cdot f(x+y+z+w)];$

Useful Observations About This Splitting

Observation 1

Any two coefficients $\alpha \neq \beta \in \{0,1\}^n$ will be in different buckets **with high probability.**



Observation 2 - Union bound

We need **small** number of random linear functions $r_1, \ldots, r_s$ to separate **all of**, say, top 3 Fourier coefficients.

Parseval's identity

$\sum_\gamma \hat{f}(\gamma)^2 = 1.$

Observation 3

1. $\sum_{\gamma \in V+h} \hat{f}(\gamma)^4 = \mathbf{E}_{\substack{x,y,z \sim \{0,1\}^n \\ w \sim V^\perp}} [\chi_h(w) \cdot f(x) \cdot f(y) \cdot f(z) \cdot f(x+y+z+w)]$;
2. $\left(\sum_{\gamma \in V+h} \hat{f}(\gamma)^4\right)^{1/4}$ is close to the *maximum* coefficients in $V + h$.

Algorithm for Computing Top t Coefficient Values

Algorithm for Computing Top t Coefficient Values

Theorem

There is an algorithm such that given (membership) query access to a function $f : \{0,1\}^n \to \{0,1\}$, and parameters $t \in \mathbb{N}, \varepsilon > 0$, makes $\text{poly}(t, 1/\varepsilon)$ queries to f and outputs t real numbers $c_1, \ldots, c_t \in \mathbb{R}^+$ that correspond to top t Fourier coefficient values of f.

Finding Largest t Coefficients

Task - Non-Interactive

Input Oracle $f : \{0,1\}^n \to \{0,1\}$, and $t \in \mathbb{N}, \varepsilon > 0$
Output $c_1, \ldots, c_t \in \mathbb{R}^+$ the absolute value of "largest" Fourier coefficients of f

Finding Largest t Coefficients

Task - Non-Interactive

Input Oracle $f : \{0,1\}^n \to \{0,1\}$, and $t \in \mathbb{N}, \varepsilon > 0$
Output $c_1, \dots, c_t \in \mathbb{R}^+$ the absolute value of "largest" Fourier coefficients of f

Algorithm

Let $\Lambda_t = \{\gamma_1, \dots, \gamma_t\}$ be the correct set.

1. Split the domain $\{0,1\}^n$ to affine subspaces $V_0, \dots, V_{2^s-1}$.
   All γ_i's belong to different cosets w.h.p.; Here $s \approx \log(t + 1/\varepsilon)$.
2. Estimate the largest coefficient in each V_i
3. Output the largest t

Finding Largest t Coefficients

Task - Non-Interactive

Input Oracle $f : \{0,1\}^n \to \{0,1\}$, and $t \in \mathbb{N}, \varepsilon > 0$
Output $c_1, \ldots, c_t \in \mathbb{R}^+$ the absolute value of "largest" Fourier coefficients of f

Algorithm

Let $\Lambda_t = \{\gamma_1, \ldots, \gamma_t\}$ be the correct set.

1. Split the domain $\{0,1\}^n$ to affine subspaces $V_0, \ldots, V_{2^s-1}$.
   All γ_i's belong to different cosets w.h.p.; Here $s \approx \log(t + 1/\varepsilon)$.
2. Estimate the largest coefficient in each V_i
3. Output the largest t

- The query complexity is poly$(t, 1/\varepsilon)$, *independent of n*;
- The characters γ_i's are still unknown!

IP for Finding Top t Fourier Coefficients

IP for Finding Top t Fourier Coefficients

Theorem

There is an interactive protocol for finding top t Fourier coefficients of a function $f : \{0,1\}^n \to \{0,1\}$ with error parameter ε, where the Verifier uses $\text{poly}(t, 1/\varepsilon)$ *membership queries*, *independent of n*.

Use Prover to Find γ_i's

Task - Interactive

Input Oracle $f : \{0,1\}^n \to \{0,1\}$, and $t \in \mathbb{N}, \varepsilon > 0$

Output $\gamma_1, \ldots, \gamma_t \in \{0,1\}^n$ the characters corresponding to the "largest" t Fourier coefficients of f

Use Prover to Find γ_i's

Task - Interactive

Input Oracle $f : \{0,1\}^n \to \{0,1\}$, and $t \in \mathbb{N}, \varepsilon > 0$
Output $\gamma_1, \ldots, \gamma_t \in \{0,1\}^n$ the characters corresponding to the "largest" t Fourier coefficients of f

Interactive Protocol - poly$(t, 1/\varepsilon)$ samples

P Sends a set $\Lambda'_t = \{\gamma'_1, \ldots, \gamma'_t\}$ of **large** coefficients

V Estimates the coefficients $c'_1, \ldots, c'_t$

V Splits the domain into $V_0, \ldots, V_{2^s-1}$
W.h.p. all of γ'_i and γ_i are separated;

V Reject if $\gamma'_i s$ cannot be matched with high-weight cosets.

Use Prover to Find γ_i's

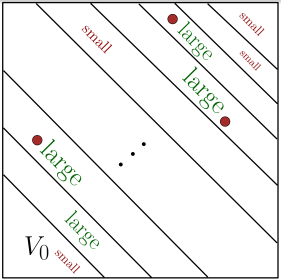
Task - Interactive

Input Oracle $f : \{0,1\}^n \to \{0,1\}$, and $t \in \mathbb{N}, \varepsilon > 0$

Output $\gamma_1, \ldots, \gamma_t \in \{0,1\}^n$ the characters corresponding to the "largest" t Fourier coefficients of f

Interactive Protocol - poly$(t, 1/\varepsilon)$ samples

- **P** Sends a set $\Lambda'_t = \{\gamma'_1, \ldots, \gamma'_t\}$ of **large** coefficients
- **V** Estimates the coefficients $c'_1, \ldots, c'_t$
- **V** Splits the domain into $V_0, \ldots, V_{2^s-1}$
  W.h.p. all of γ'_i and γ_i are separated;
- **V** Reject if $\gamma'_i s$ cannot be matched with high-weight cosets.



Honest Prover

Use Prover to Find γ_i's

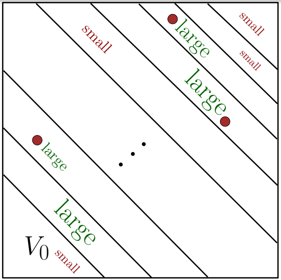
Task - Interactive

Input Oracle $f : \{0,1\}^n \to \{0,1\}$, and $t \in \mathbb{N}, \varepsilon > 0$

Output $\gamma_1, \ldots, \gamma_t \in \{0,1\}^n$ the characters corresponding to the "largest" t Fourier coefficients of f

Interactive Protocol - poly$(t, 1/\varepsilon)$ samples

- **P** Sends a set $\Lambda'_t = \{\gamma'_1, \ldots, \gamma'_t\}$ of **large** coefficients
- **V** Estimates the coefficients $c'_1, \ldots, c'_t$
- **V** Splits the domain into $V_0, \ldots, V_{2^s-1}$
  W.h.p. all of γ'_i and γ_i are separated;
- **V** Reject if $\gamma'_i s$ cannot be matched with high-weight cosets.



Cheating Prover

IP for Finding Top t Fourier Coefficients

Theorem

There is an interactive protocol for finding top t Fourier coefficients of a function $f : \{0,1\}^n \to \{0,1\}$ with error parameter ε, where the Verifier uses $\text{poly}(t, 1/\varepsilon)$ *membership queries*, *independent of n*.

Can we use random examples instead?

Our Result

Main Theorem - Random Examples

There is an interactive protocol for finding top t Fourier coefficients of a function $f : \{0,1\}^n \to \{0,1\}$ with error parameter ε, where the Verifier uses $\text{poly}(t, 1/\varepsilon)$ *random examples*, *independent of n*.

Query-to-Sample Reduction

Random Example

$(x, f(x))$ where $x \sim \{0,1\}^n$.

Query-to-Sample Reduction

Random Example

$(x, f(x))$ where $x \sim \{0,1\}^n$.

General Framework of Query-to-Sample Reduction

Query-to-Sample Reduction

Random Example

$(x, f(x))$ where $x \sim \{0,1\}^n$.

General Framework of Query-to-Sample Reduction

- Idea: Prover will answer Verifier's membership queries.

Query-to-Sample Reduction

Random Example

$(x, f(x))$ where $x \sim \{0,1\}^n$.

General Framework of Query-to-Sample Reduction

- Idea: Prover will answer Verifier's membership queries.
- P_M: IP with q *membership* queries $\rightarrow$ P_R: IP with $O(q)$ *random examples*

Query-to-Sample Reduction

Random Example

$(x, f(x))$ where $x \sim \{0,1\}^n$.

General Framework of Query-to-Sample Reduction

- Idea: Prover will answer Verifier's membership queries.
- P_M: IP with q *membership* queries $\rightarrow$ P_R: IP with $O(q)$ *random examples*
- Requirement: P_M is MA-like

Query-to-Sample Reduction

Random Example

$(x, f(x))$ where $x \sim \{0,1\}^n$.

General Framework of Query-to-Sample Reduction

- Idea: Prover will answer Verifier's membership queries.
- P_M: IP with q *membership* queries $\to$ P_R: IP with $O(q)$ *random examples*
- Requirement: P_M is MA-like

Recall

$$\sum_{\gamma \in V+h} \hat{f}(\gamma)^4 = \mathop{\mathbf{E}}_{\substack{x,y,z \sim \{0,1\}^n \\ w \sim V^\perp}} [\chi_h(w) \cdot f(x) \cdot f(y) \cdot f(z) \cdot f(x+y+z+w)].$$

Query-to-Sample Reduction

Random Example

$(x, f(x))$ where $x \sim \{0,1\}^n$.

General Framework of Query-to-Sample Reduction

- Idea: Prover will answer Verifier's membership queries.
- P_M: IP with q *membership* queries $\to$ P_R: IP with $O(q)$ *random examples*
- Requirement: P_M is MA-like

$$\mathop{\mathbf{E}}_{\substack{x,y,z \sim \{0,1\}^n \\ w \sim V^{\perp}}} [\chi_h(w) \cdot f(\underbrace{x}_{query}) \cdot f(\underbrace{y}_{query}) \cdot f(\underbrace{z}_{query}) \cdot f(\underbrace{x+y+z+w}_{query})].$$

Query-to-Sample Reduction

Random Example

$(x, f(x))$ where $x \sim \{0,1\}^n$.

General Framework of Query-to-Sample Reduction

- Idea: Prover will answer Verifier's membership queries.
- P_M: IP with q *membership* queries $\rightarrow$ P_R: IP with $O(q)$ *random examples*
- Requirement: P_M is MA-like

Observation

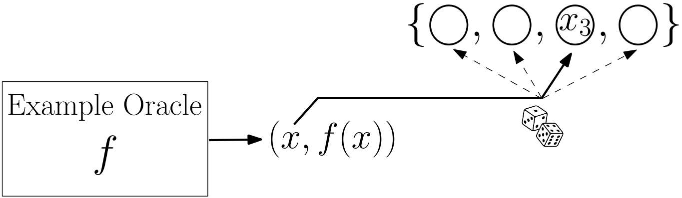
$$\mathop{\mathbf{E}}_{\substack{x,y,z\sim\{0,1\}^n \\ w\sim V^{\perp}}}[\chi_h(w)\cdot f(\underbrace{x}_{\textit{uniform}})\cdot f(\underbrace{y}_{\textit{uniform}})\cdot f(\underbrace{z}_{\textit{uniform}})\cdot f(\underbrace{x+y+z+w}_{\textit{uniform}})].$$

Embedding a Random Example

- Our query set is $\{x, y, z, x+y+z+w\} \sim \mathcal{D}$.

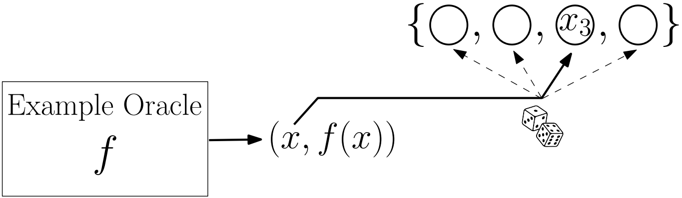
Embedding a Random Example

- Our query set is $\{x, y, z, x + y + z + w\} \sim \mathcal{D}$.



Embedding a Random Example

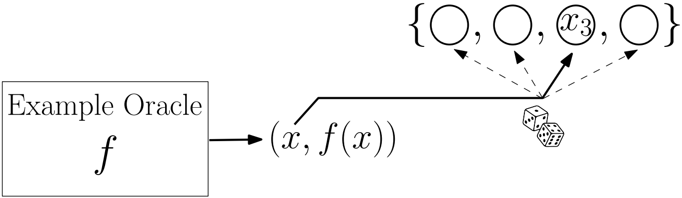
- Our query set is $\{x, y, z, x+y+z+w\} \sim \mathcal{D}$.



- We extend $\{\bigcirc, \bigcirc, x_3, \bigcirc\}$ to $\{x_1, x_2, x_3, x_4\} \sim \mathcal{D}$.

Embedding a Random Example

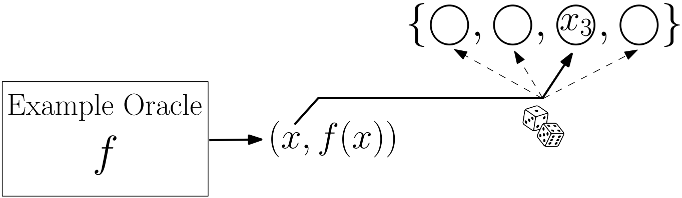
- Our query set is $\{x, y, z, x + y + z + w\} \sim \mathcal{D}$.



- We extend $\{\bigcirc, \bigcirc, x_3, \bigcirc\}$ to $\{x_1, x_2, x_3, x_4\} \sim \mathcal{D}$.
- Note that the Verifier knows $f(x_3)$.

Embedding a Random Example

- Our query set is $\{x, y, z, x+y+z+w\} \sim \mathcal{D}$.

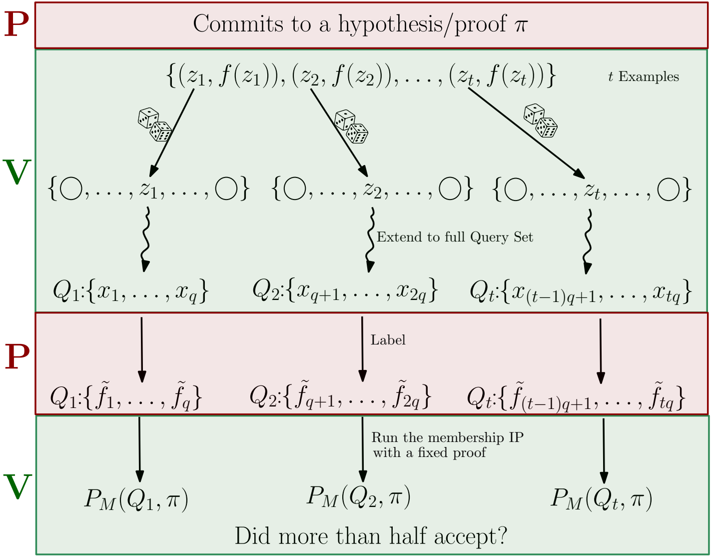


- We extend $\{\bigcirc, \bigcirc, x_3, \bigcirc\}$ to $\{x_1, x_2, x_3, x_4\} \sim \mathcal{D}$.
- Note that the Verifier knows $f(x_3)$.
- Now, if the Prover cheats in labeling this set, the Verifier catches him with probability $1/4$.

Query-to-Sample Reduction: Details

- P_M: MA-like *membership protocol*;
- π: The proof (hypothesis) sent by the prover;
- q: Number of Verifier's queries.
- Let $t = O(q)$, large enough

Query-to-Sample Reduction: Details



Interactive Proofs for Learning top Fourier Coefficients

Main Theorem

There is an interactive protocol for finding top t Fourier coefficients of a function $f : \{0,1\}^n \to \{0,1\}$ with error parameter ε, where the Verifier uses poly$(t, 1/\varepsilon)$ *random examples*, *independent of n*.

Interactive Proofs for Learning top Fourier Coefficients

Main Theorem

There is an interactive protocol for finding top t Fourier coefficients of a function $f : \{0,1\}^n \to \{0,1\}$ with error parameter ε, where the Verifier uses $\text{poly}(t, 1/\varepsilon)$ *random examples*, *independent of n*.

We have also other results

Interactive Proofs for Learning top Fourier Coefficients

Main Theorem

There is an interactive protocol for finding top t Fourier coefficients of a function $f : \{0,1\}^n \to \{0,1\}$ with error parameter ε, where the Verifier uses $\text{poly}(t, 1/\varepsilon)$ *random examples*, *independent of n*.

We have also other results

- Learning k-juntas

Interactive Proofs for Learning top Fourier Coefficients

Main Theorem

There is an interactive protocol for finding top t Fourier coefficients of a function $f : \{0,1\}^n \to \{0,1\}$ with error parameter ε, where the Verifier uses $\mathrm{poly}(t, 1/\varepsilon)$ *random examples*, *independent of n*.

We have also other results

- Learning k-juntas
- Learning $\mathsf{AC}^0[\bigoplus]$

Interactive Proofs for Learning top Fourier Coefficients

Main Theorem

There is an interactive protocol for finding top t Fourier coefficients of a function $f : \{0,1\}^n \to \{0,1\}$ with error parameter ε, where the Verifier uses $\text{poly}(t, 1/\varepsilon)$ *random examples*, *independent of n*.

We have also other results

- Learning k-juntas
- Learning $\mathrm{AC}^0[\bigoplus]$
- Some results for arbitrary classes

Thank you for listening!