Solving Noisy k-XOR below the $n^{k/2}$ threshold

Arpon Basu   Jun-Ting Hsieh   Andrew D. Lin   Peter Manohar

Noisy k-XOR Problem

- Given random k-uniform hypergraph $\mathcal{H}$ on $[n]$
- For every **clause** C in $\mathcal{H}$: equation $\prod_{i\in C} x_i = \prod_{i\in C} x_i^*$
- **Planted** solution $x^* \in \{-1,1\}^n$ satisfies equations
- Flip each RHS independently w.p. 49%

Goal: Find planted assignment x^*

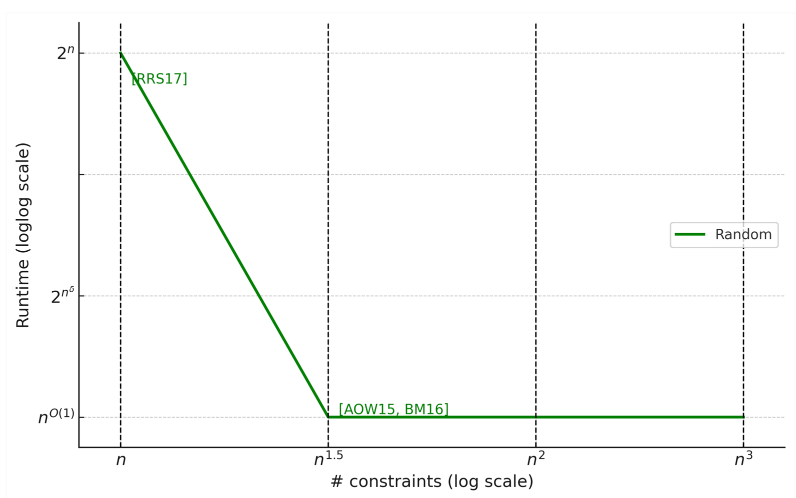
- $k = 2$ corresponds to **Stochastic Block Model**
- k-noisy XOR a.k.a. k-sparse **Learning Parity with Noise**

Prior Work

Feldman-Perkins-Vempala'15: Can recover x^* in poly(n) time if $\geqslant \widetilde{\Omega}(n^{k/2})$ clauses

Subexponential tradeoff for refutation:

- **Raghavendra-Rao-Schramm'17:** With $\gtrsim n(n/\ell)^{k/2-1}$ clauses, can *refute* random k-XOR in $n^{O(\ell)}$ time. Trade-off for $k=3$ below



Open: Subexponential tradeoff for *planted* CSPs?

Our Main Result

> **Theorem:** Can solve random planted k-XOR/break k-sparse LPN with
> $m \gtrsim n(n/\ell)^{k/2-1}$ clauses in $n^{O(\ell)}$ time.

Correct analog of RRS'17 to the planted CSP setting!

Can solve all random CSPs!

Given CSP predicate $P : \{-1,1\}^k \to \{0,1\}$, we can reduce random planted CSP to noisy XOR by Fourier analysis on P.

Consequence: Given random k-CSP with $\gtrsim n(n/\ell)^{k/2-1}$ clauses, can find satisfying assignment (assuming P has one) in $n^{O(\ell)}$ time

Our Algorithm

Two-step approach:

1. **Find approximate solution** using Sum-of-Squares
2. **Round to exact solution** using local improvement

Both parts crucially use randomness of hypergraph $\mathcal{H}$

Canonical Sum-of-Squares Program

Define objective function: $\psi(x) = \mathbf{E}_{C\sim\mathcal{H}}[b_C \cdot x_C]$ where b_C is RHS of equation for clause C, and $x_C = \prod_{i\in C} x_i$

Canonical SoS Relaxation: Maximize $\psi(x)$ using deg ℓ SoS

deg ℓ SoS is a polynomial optimization algorithm which runs in $n^{O(\ell)}$ time

Step 1: Approximate Solution via SoS

Key property: Since $\mathcal{H}$ is random,

$$\psi(x) = \mathbf{E}_{C\sim\mathcal{H}}[b_C \cdot x_C] \approx \langle x, x^*\rangle^k$$

for all x.

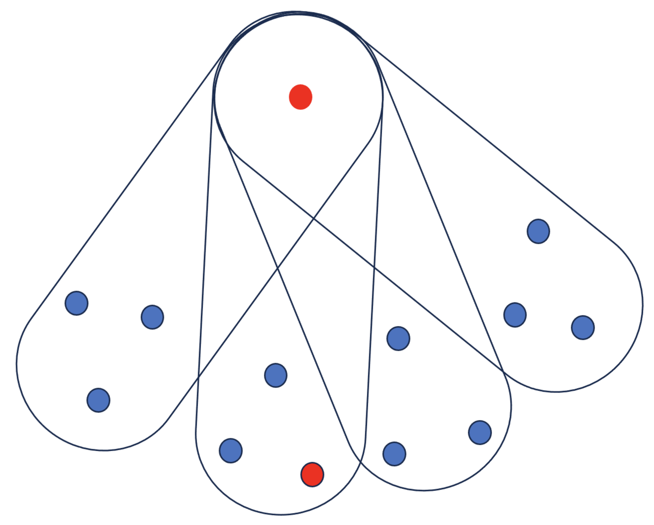
Consequences of Randomness:

- $\psi(x)$ maximized at $x = x^*$
- Degree-ℓ Sum-of-Squares recognizes this!
- SoS finds solution x that is $1-o(1)$ correlated with x^*

Step 2: Rounding via Local Improvement

Setup: We have x that is $1-o(1)$ correlated with x^*
Let "bad" = set of indices $i \in [n]$ where x, x^* differ

Key observation: Because $\mathcal{H}$ is random, a clause C containing a bad index doesn't contain any other bad index w.p. $\geqslant 1-o(1)$



Recovery: If i is bad and C containing i has no other bad index: $b_C = x_i \cdot x_{C\setminus\{i\}} = x_i \cdot x^*_{C\setminus\{i\}} \implies x_i = b_C \cdot x_{C\setminus\{i\}} = x_i^*$
With $\log n$ clauses containing i, majority vote recovers x_i^* w.h.p.

Future Directions

- Extend to **semi-random CSPs**: Variables in clauses are arbitrary, literals are random
- No known guarantees on the performance of canonical SoS program on semi-random instance!
- Guruswami-Hsieh-Kothari-Manohar'23 can solve semi-random planted CSPs when $m \gtrsim n^{k/2}$, but subexp tradeoff not known

Find our Paper!

Full paper:
`arxiv.org/abs/2507.10833`
Scan the QR code for the paper