On the Hardness of the One-Sided Code Sparsifier Problem

Alice Moayyedi

University of Waterloo

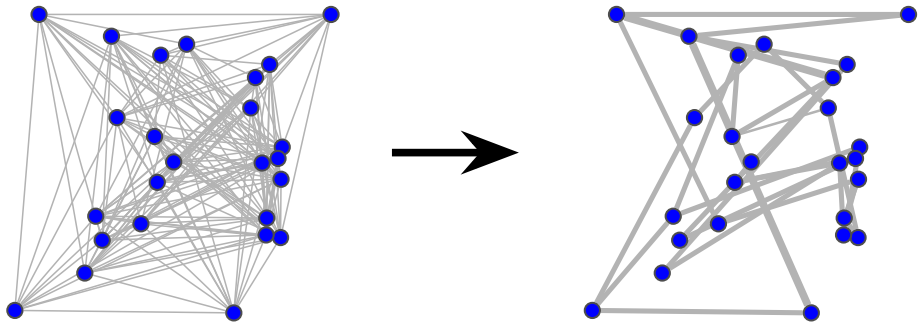
March 10, 2026

Joint work with Elena Grigorescu (University of Waterloo)

Overview

Sparsifiers

- Given an object S;
- Wish to produce a **sparse** object S' which approximately preserves some attributes of S
  - Examples: cuts of graphs, assignments of SAT instances, codewords of codes
- Two-sided sparsifier: for all attributes p, $(1-\epsilon)p(S') \leq p(S) \leq (1+\epsilon)p(S')$
- One-sided sparsifier: $p(S) \leq p(S')$
- Unweighted versions: $(1-\epsilon)p(S') \leq \alpha \cdot p(S) \leq (1+\epsilon)p(S')$; $\alpha \cdot p(S) \leq p(S')$



S (dense) **S′** (sparse, maintains structure)

Codes

- Linear code: subspace $\mathcal{C} \subseteq \mathbb{F}_2^n$ $(a + b \in \mathcal{C}; \forall a, b \in \mathcal{C})$

Codes

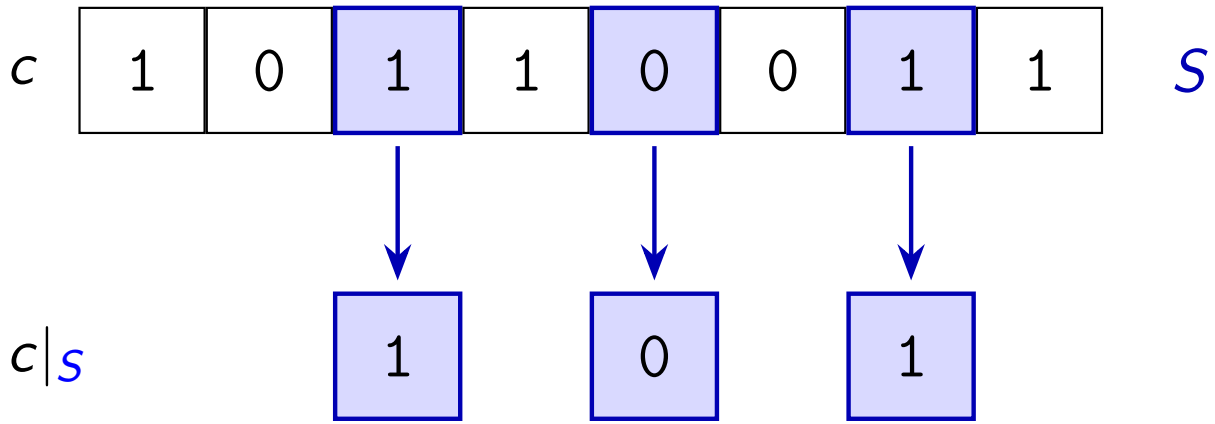
- Linear code: subspace $\mathcal{C} \subseteq \mathbb{F}_2^n$ $(a + b \in \mathcal{C}; \forall a, b \in \mathcal{C})$
- Word: string in $\mathbb{F}_2^n$
- Codeword: string in $\mathcal{C}$

Codes

- Linear code: subspace $\mathcal{C} \subseteq \mathbb{F}_2^n$ $(a + b \in \mathcal{C}; \forall a, b \in \mathcal{C})$
- Word: string in $\mathbb{F}_2^n$
- Codeword: string in $\mathcal{C}$
- Hamming weight: $\mathrm{wt}(c) =$ number of coordinates of c which equal 1
- Hamming distance: $\mathrm{d}(a, b) =$ number of coordinates where a and b have different values

Codes

- Linear code: subspace $\mathcal{C} \subseteq \mathbb{F}_2^n$ $(a + b \in \mathcal{C}; \forall a, b \in \mathcal{C})$
- Word: string in $\mathbb{F}_2^n$
- Codeword: string in $\mathcal{C}$
- Hamming weight: $\mathrm{wt}(c) =$ number of coordinates of c which equal 1
- Hamming distance: $\mathrm{d}(a, b) =$ number of coordinates where a and b have different values
- α-sparsifier: set $S \in [n]$ of coordinates s.t. $\mathrm{wt}_S(c) \geq \alpha \cdot \mathrm{wt}(c); \forall c \in \mathcal{C}$
  - The smaller the sparsifier, the better



$$\mathrm{wt}(c) = 5,\ \mathrm{wt}_S(c) = 2$$

(Linear) Code Sparsification

- Do small weighted code sparsifiers exist?

(Linear) Code Sparsification

- Do small weighted code sparsifiers exist?
- Yes. [Khanna, Putterman, and Sudan, 2024]

(Linear) Code Sparsification

- Do small weighted code sparsifiers exist?
- Yes. [Khanna, Putterman, and Sudan, 2024]
- Can we find them efficiently?

(Linear) Code Sparsification

- Do small weighted code sparsifiers exist?
- Yes. [Khanna, Putterman, and Sudan, 2024]
- Can we find them efficiently?
- Yes. [Khanna, Putterman, and Sudan, 2024]

(Linear) Code Sparsification

- Do small weighted code sparsifiers exist?
- Yes. [Khanna, Putterman, and Sudan, 2024]
- Can we find them efficiently?
- Yes. [Khanna, Putterman, and Sudan, 2024]
- Do small unweighted code sparsifiers exist?

(Linear) Code Sparsification

- Do small weighted code sparsifiers exist?
- Yes. [Khanna, Putterman, and Sudan, 2024]
- Can we find them efficiently?
- Yes. [Khanna, Putterman, and Sudan, 2024]
- Do small unweighted code sparsifiers exist?
- Yes. [Oveis Gharan and Sahami, 2025]

(Linear) Code Sparsification

- Do small weighted code sparsifiers exist?
- Yes. [Khanna, Putterman, and Sudan, 2024]
- Can we find them efficiently?
- Yes. [Khanna, Putterman, and Sudan, 2024]
- Do small unweighted code sparsifiers exist?
- Yes. [Oveis Gharan and Sahami, 2025]
- Can we find them efficiently?

(Linear) Code Sparsification

- Do small weighted code sparsifiers exist?
- Yes. [Khanna, Putterman, and Sudan, 2024]
- Can we find them efficiently?
- Yes. [Khanna, Putterman, and Sudan, 2024]
- Do small unweighted code sparsifiers exist?
- Yes. [Oveis Gharan and Sahami, 2025]
- Can we find them efficiently?
- No! (This talk!)

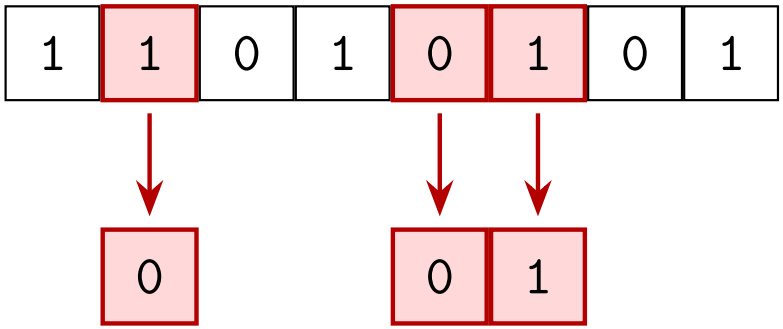
Main Theorem

The following problem is NP-hard:

OptSparsifier

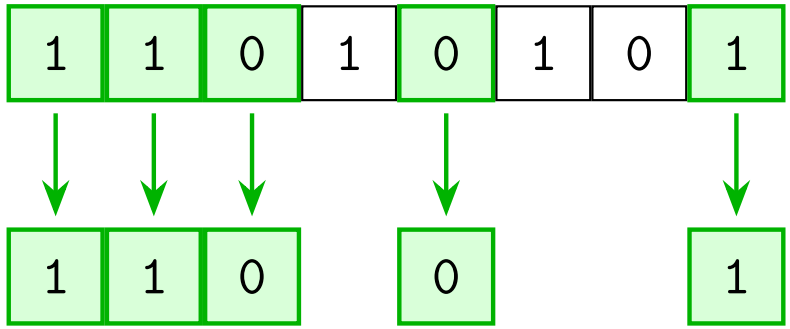
Instance: A linear code $\mathcal{C} \subseteq \mathbb{F}_2^n$ such that $\mathbf{1} \in \mathcal{C}$ (the all-ones codeword)
Output: A set $S \subseteq [n]$ such that:

- **(Feasibility)** for all $c \in \mathcal{C}$, $\mathrm{wt}(c_S) \geq \frac{1}{2}\mathrm{wt}(c)$;
- **(Optimality)** S is of smallest size among all sets that satisfy the above



$\mathrm{wt}_S(c) < \frac{1}{2}\,\mathrm{wt}(c)$

$1 < \frac{5}{2}$



$\mathrm{wt}_S(c) \geq \frac{1}{2}\,\mathrm{wt}(c)$

$3 \geq \frac{5}{2}$

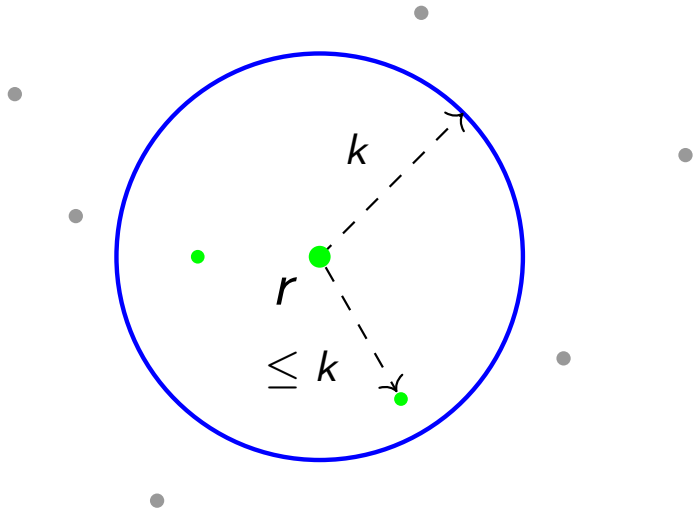
Nearest Codeword Problem

Proof will be a Turing reduction from the classical Nearest Codeword Problem (NCP) to OptSparsifier

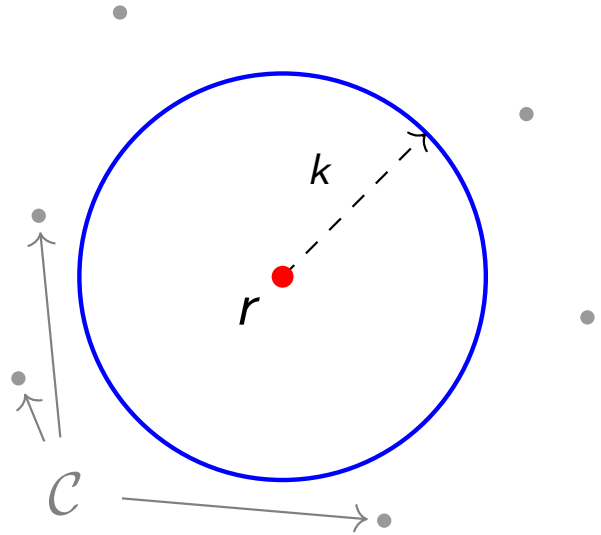
NCP

Instance: received word $r \in \mathbb{F}_2^n$ and integer k (distance)
Output: whether $\min_{c \in \mathcal{C}} d(c, r) \leq k$.

- NCP is hard and also **hard to approximate** up to any constant factor



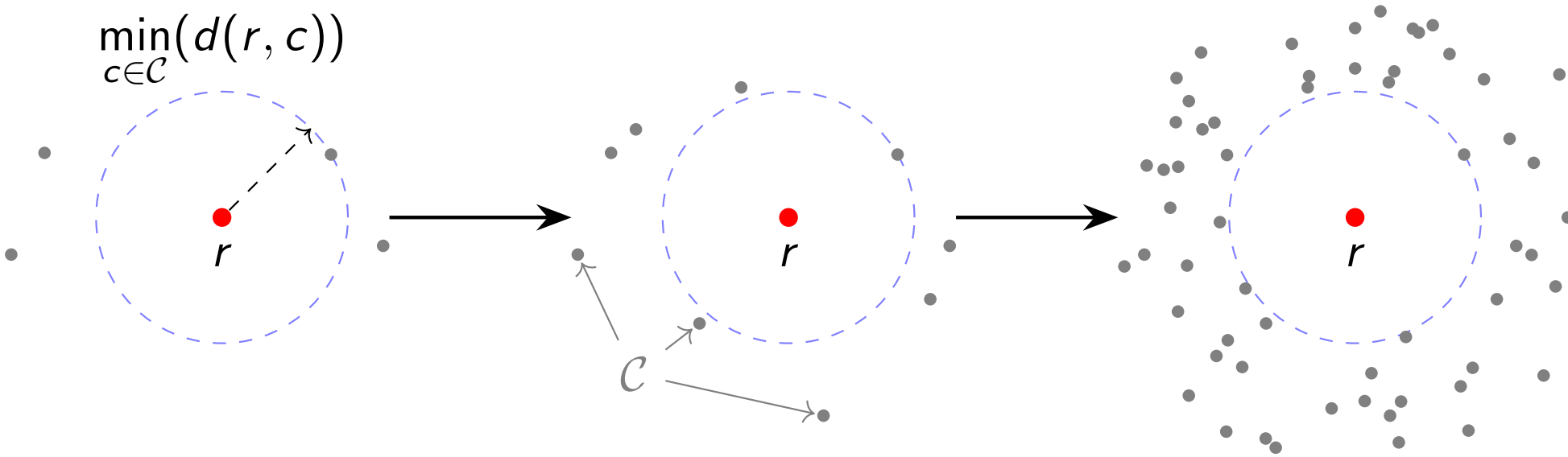
A (YES) instance of NCP



A (NO) instance of NCP

Proof Strategy

- Given a code $\mathcal{C}$, a received word r, and a distance k:
  1. enlarge the code **without moving it closer to r**;
  2. . . . until we can't any more, and then **see how far r is from the code**

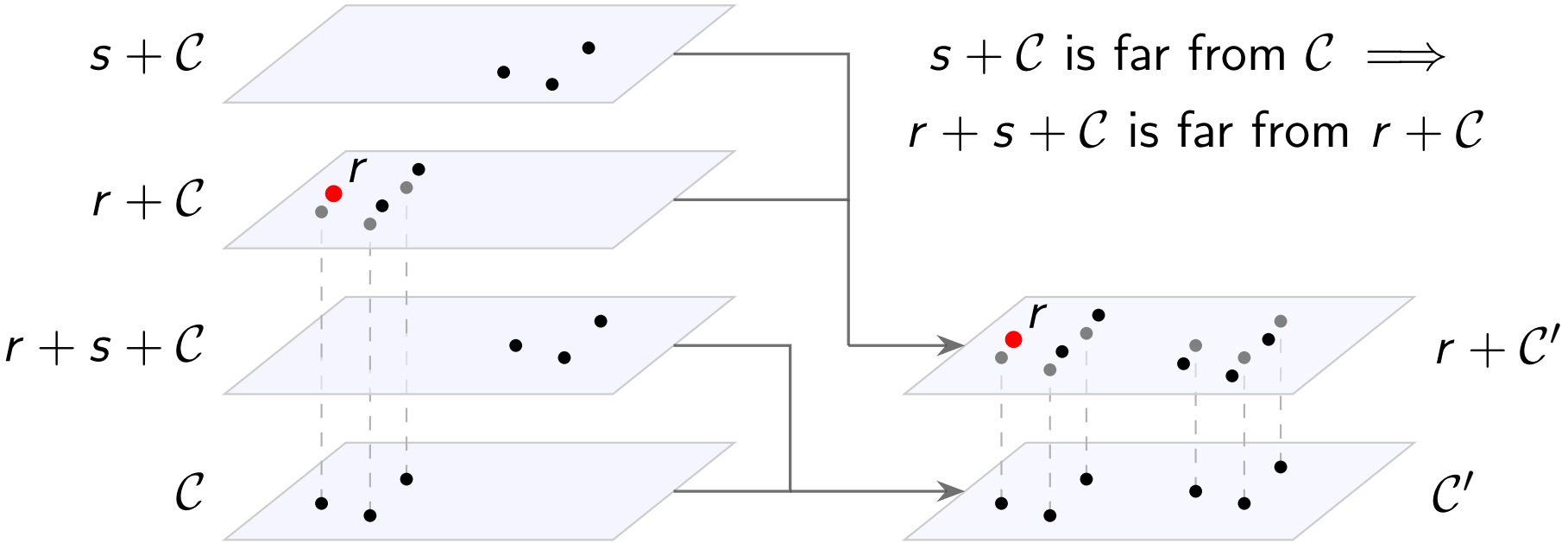


Structural Properties

- Consider sparsifiers as words, given by their indicator vectors: $s_i = 1 \iff i \in S$, $s_i = 0 \iff i \notin S$.
- **What properties do the smallest sparsifiers have?**
  1. They're among the words **furthest from the code**, and
  2. it's **easy to tell how far** ($\mathbf{1}$ is a closest codeword)
- For $s \in \mathbb{F}_2^n$, all $s + c$ for $c \in \mathcal{C}$ are equally far from $\mathcal{C}$
  3. This means that if r and s are in the same coset of $\mathbb{F}_2^n/\mathcal{C}$, then r and s are **equally far** from the code

Algorithm

- Start with NCP instance $(\mathcal{C}, r, k)$.
- Loop:
  - Using the OptSparsifier oracle, find a smallest sparsifier s for $\mathcal{C}$.
  - Check whether s **has distance** $\leq k$ to $\mathcal{C}$ (2). If so, r **does too** (1).
    - Original NCP instance is a (YES) instance.
  - Check whether $r \in s + \mathcal{C}$. If so, r **is as close to** $\mathcal{C}$ **as** s **is** (3).
    - Original NCP instance is a (NO) instance.
  - If not, **add** $s + r + \mathcal{C}$ **to** $\mathcal{C}$ (1).



Summary

- NCP is hard to approximate (assuming $P \neq NP$), so we can't efficiently get a sparsifier of almost any size!
- We can't efficiently reduce the number of coordinates by any fixed proportion (e.g., 1%) and be sure that we'll keep half of the weight

Future Work

- Proof "goes through" codes outside of any particular family — are some families also hard to sparsify? Does some family for which NCP is hard have good and efficiently-findable sparsifiers?
- Proof only works for $\alpha = 1/2$; are there nontrivial values for which a good one-sided sparsification is possible?
- One-sided sparsifiers in graphs are unstudied

Thank you for your time!