Learning without correspondence

Daniel Hsu

Computer Science Department & Data Science Institute
Columbia University

Introduction

Example #1: unlinked data sources

- Two separate data sources about same entities:

Sex	Age	Height
M	20	180
F	24	162.5
F	22	160
F	23	167.5

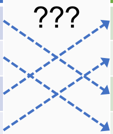
Disease
1
0
0
1

  - First source contains covariates (sex, age, height, ...).
  - Second source contains response variable (disease status).

Example #1: unlinked data sources

- Two separate data sources about same entities:

Sex	Age	Height
M	20	180
F	24	162.5
F	22	160
F	23	167.5



Disease
1
0
0
1

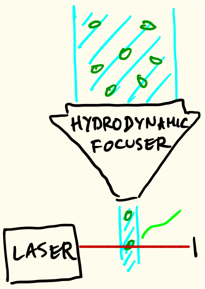
  - First source contains covariates (sex, age, height, ...).
  - Second source contains response variable (disease status).

To learn: relationship between response and covariates.

Record linkage unknown.

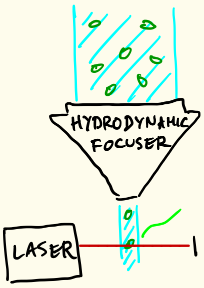
Example #2: flow cytometry

1. Suspended cells in fluid.
2. Cells pass through laser, one at a time; measure emitted light.



Example #2: flow cytometry

1. Suspended cells in fluid.
2. Cells pass through laser, one at a time; measure emitted light.

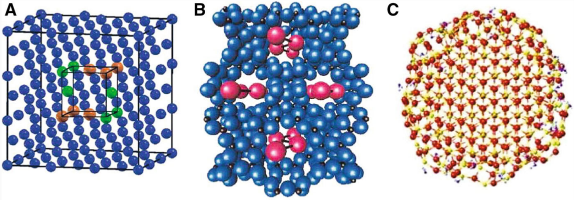


To learn: relationship between measurements and cell properties.

Order in which cells pass through laser is unknown.

Example #3: unassigned distance geometry

1. Unknown arrangement of n points in Euclidean space.

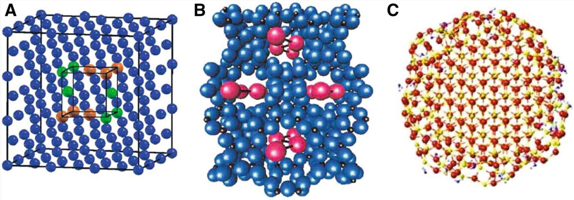


(Image credit: Billinge, Duxbury, Gonçalves, Lavor, & Mucherino, 2016)

2. Measure distribution of *pairwise distances* among the n points (using high-energy X-rays).

Example #3: unassigned distance geometry

1. Unknown arrangement of n points in Euclidean space.



(Image credit: Billinge, Duxbury, Gonçalves, Lavor, & Mucherino, 2016)

2. Measure distribution of *pairwise distances* among the n points (using high-energy X-rays).

To learn: original arrangement of the n points.

Assignment of distances to pairs of points is unknown.

Learning without correspondence

Observation:

Correspondence information is missing in many natural settings.

Learning without correspondence

Observation:

Correspondence information is missing in many natural settings.

Question:

How does this affect machine learning / statistical estimation?

Learning without correspondence

Observation:
Correspondence information is missing in many natural settings.

Question:
How does this affect machine learning / statistical estimation?

We give a theoretical treatment in context of two simple problems:

1. **Linear regression without correspondence**
   (Joint work with Kevin Shi and Xiaorui Sun; NIPS 2017.)
2. **Correspondence retrieval** (generalization of *phase retrieval*)
   (Joint work with Alexandr Andoni, Kevin Shi, and Xiaorui Sun; COLT 2017.)

Our contributions

1. **Linear regression without correspondence**
   - Strong NP-hardness of least squares problem.
   - Polynomial-time approximation scheme in constant dimensions.
   - Information-theoretic signal-to-noise lower bounds.
   - Polynomial-time algorithm in noise-free average case setting.
2. **Correspondence retrieval**
   - Measurement-optimal recovery algorithm in noise-free setting.
   - Robust recovery algorithm in noisy setting.

Linear regression without correspondence

Linear regression without correspondence

y_1	$\boldsymbol{x}_1^\top$
y_2	$\boldsymbol{x}_2^\top$
$\vdots$	$\vdots$
y_n	$\boldsymbol{x}_n^\top$

Feature vectors: $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$

Labels: $y_1, y_2, \ldots, y_n \in \mathbb{R}$

Linear regression without correspondence

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_1^\top \\ \boldsymbol{x}_2^\top \\ \vdots \\ \boldsymbol{x}_n^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}^* \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

Classical linear regression:

$$y_i = \boldsymbol{x}_i^\top \boldsymbol{\beta}^* + \varepsilon_i, \quad i = 1, \ldots, n.$$

Linear regression without correspondence

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_{\pi^*(1)}^\top \\ \boldsymbol{x}_{\pi^*(2)}^\top \\ \vdots \\ \boldsymbol{x}_{\pi^*(n)}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}^* \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

Linear regression without correspondence:

$$y_i = \boldsymbol{x}_{\pi^*(i)}^\top \boldsymbol{\beta}^* + \varepsilon_i, \quad i = 1, \ldots, n.$$

Model for linear regression without correspondence

Unnikrishnan, Haghighatshoar, & Vetterli, 2015; Pananjady, Wainwright, & Courtade 2016; Elhami, Scholefield, Haro, & Vetterli, 2017; Abid, Poon, & Zou, 2017; ...

- **Feature vectors**: $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$
- **Labels**: $y_1, y_2, \ldots, y_n \in \mathbb{R}$
- **Model**:

$$y_i \;=\; \boldsymbol{x}_{\pi^*(i)}^\top \boldsymbol{\beta}^* + \varepsilon_i\,, \quad i = 1, \ldots, n.$$

  - Linear function: $\boldsymbol{\beta}^* \in \mathbb{R}^d$
  - Permutation: $\pi^* \in S_n$
  - Errors: $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n \in \mathbb{R}$.

Model for linear regression without correspondence

Unnikrishnan, Haghighatshoar, & Vetterli, 2015; Pananjady, Wainwright, & Courtade 2016; Elhami, Scholefield, Haro, & Vetterli, 2017; Abid, Poon, & Zou, 2017; ...

- **Feature vectors**: $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$
- **Labels**: $y_1, y_2, \ldots, y_n \in \mathbb{R}$
- **Model**:

$$y_i \;=\; \boldsymbol{x}_{\pi^*(i)}^\top \boldsymbol{\beta}^* + \varepsilon_i \,, \quad i = 1, \ldots, n.$$

  - Linear function: $\boldsymbol{\beta}^* \in \mathbb{R}^d$
  - Permutation: $\pi^* \in S_n$
  - Errors: $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n \in \mathbb{R}$.
- **Goal**: "learn" $\boldsymbol{\beta}^*$.

Model for linear regression without correspondence

Unnikrishnan, Haghighatshoar, & Vetterli, 2015; Pananjady, Wainwright, & Courtade 2016; Elhami, Scholefield, Haro, & Vetterli, 2017; Abid, Poon, & Zou, 2017; ...

- **Feature vectors**: $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$
- **Labels**: $y_1, y_2, \ldots, y_n \in \mathbb{R}$
- **Model**:

$$y_i \ = \ \boldsymbol{x}_{\pi^*(i)}^\top \boldsymbol{\beta}^* + \varepsilon_i \,, \quad i = 1, \ldots, n.$$

  - Linear function: $\boldsymbol{\beta}^* \in \mathbb{R}^d$
  - Permutation: $\pi^* \in S_n$
  - Errors: $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n \in \mathbb{R}$.
- **Goal**: "learn" $\boldsymbol{\beta}^*$.

Correspondence between $(\boldsymbol{x}_i)_{i=1}^n$ and $(y_i)_{i=1}^n$ is **unknown**.

Questions

1. Can we determine if there is a good linear fit to the data? (Least squares approximation.)

Questions

1. Can we determine if there is a good linear fit to the data?
   (Least squares approximation.)
2. When is it possible to recover the "correct" $\boldsymbol{\beta}^*$?
   (When is the "best" linear fit actually meaningful?)

Least squares approximation

Least squares problem

Given $(\boldsymbol{x}_i)_{i=1}^n$ from $\mathbb{R}^d$ and $(y_i)_{i=1}^n$ from $\mathbb{R}$, minimize

$$F(\boldsymbol{\beta}, \pi) \ := \ \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{\beta} - y_{\pi(i)} \right)^2 .$$

Least squares problem

Given $(\boldsymbol{x}_i)_{i=1}^n$ from $\mathbb{R}^d$ and $(y_i)_{i=1}^n$ from $\mathbb{R}$, minimize

$$F(\boldsymbol{\beta}, \pi) \;:=\; \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{\beta} - y_{\pi(i)} \right)^2 .$$

- $d = 1$: $O(n \log n)$-time algorithm.

  (Observed by Pananjady, Wainwright, & Courtade, 2016.)

Least squares problem

Given $(\boldsymbol{x}_i)_{i=1}^n$ from $\mathbb{R}^d$ and $(y_i)_{i=1}^n$ from $\mathbb{R}$, minimize

$$F(\boldsymbol{\beta}, \pi) := \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{\beta} - y_{\pi(i)} \right)^2 .$$

- $d = 1$: $O(n \log n)$-time algorithm.

  (Observed by Pananjady, Wainwright, & Courtade, 2016.)

- $d = \Omega(n)$: (strongly) NP-hard to decide if $\min F = 0$.
  Reduction from 3-PARTITION (H., Shi, & Sun, 2017).

Least squares problem

Given $(\boldsymbol{x}_i)_{i=1}^n$ from $\mathbb{R}^d$ and $(y_i)_{i=1}^n$ from $\mathbb{R}$, minimize

$$F(\boldsymbol{\beta}, \pi) \ := \ \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{\beta} - y_{\pi(i)} \right)^2 .$$

- $d = 1$: $O(n \log n)$-time algorithm.

  (Observed by Pananjady, Wainwright, & Courtade, 2016.)

- $d = \Omega(n)$: (strongly) NP-hard to decide if $\min F = 0$.
  Reduction from 3-PARTITION (H., Shi, & Sun, 2017).

Naïve brute-force search: $\Omega(|S_n|) = \Omega(n!)$.

Least squares problem

Given $(\boldsymbol{x}_i)_{i=1}^n$ from $\mathbb{R}^d$ and $(y_i)_{i=1}^n$ from $\mathbb{R}$, minimize

$$F(\boldsymbol{\beta}, \pi) \ := \ \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{\beta} - y_{\pi(i)} \right)^2 .$$

- $d = 1$: $O(n \log n)$-time algorithm.

  (Observed by Pananjady, Wainwright, & Courtade, 2016.)

- $d = \Omega(n)$: (strongly) NP-hard to decide if $\min F = 0$.
  Reduction from 3-PARTITION (H., Shi, & Sun, 2017).

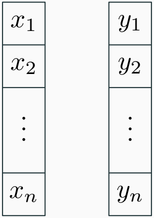
Naïve brute-force search: $\Omega(|S_n|) = \Omega(n!)$.

Least squares with known correspondence: $O(nd^2)$ time.

Least squares problem ($d = 1$)

Given $(\boldsymbol{x}_i)_{i=1}^n$ and $(y_i)_{i=1}^n$ from $\mathbb{R}$, minimize

$$F(\beta, \pi) \;:=\; \sum_{i=1}^{n} \left(x_i \beta - y_{\pi(i)} \right)^2 .$$



Least squares problem ($d = 1$)

Given $(\boldsymbol{x}_i)_{i=1}^n$ and $(y_i)_{i=1}^n$ from $\mathbb{R}$, minimize

$$F(\beta, \pi) \; := \; \sum_{i=1}^{n} \left(x_i \beta - y_{\pi(i)} \right)^2 .$$

$$\begin{array}{l} \left(x_1\beta \quad - \quad y_1\right)^2 \\ \left(x_2\beta \quad - \quad y_2\right)^2 \\ \quad \vdots \qquad\quad \vdots \\ \left(x_n\beta \quad - \quad y_n\right)^2 \end{array}$$

Cost with $\pi(i) = i$ for all $i = 1, \ldots, n$.

Least squares problem ($d = 1$)

Given $(\boldsymbol{x}_i)_{i=1}^n$ and $(y_i)_{i=1}^n$ from $\mathbb{R}$, minimize

$$F(\beta, \pi) \ := \ \sum_{i=1}^{n} \left(x_i \beta - y_{\pi(i)} \right)^2 .$$

$$(3\beta - 2)^2$$
$$(4\beta - 1)^2$$
$$\vdots \qquad \vdots$$
$$(6\beta - 7)^2$$

Cost with $\pi(i) = i$ for all $i = 1, \ldots, n$.

Least squares problem ($d = 1$)

Given $(\boldsymbol{x}_i)_{i=1}^n$ and $(y_i)_{i=1}^n$ from $\mathbb{R}$, minimize

$$F(\beta, \pi) \ := \ \sum_{i=1}^{n} \left(x_i \beta - y_{\pi(i)} \right)^2 .$$

$$\begin{array}{l} (3\beta - 2)^2 \\ (4\beta - 1)^2 \\ \quad\vdots \qquad \vdots \\ (6\beta - 7)^2 \end{array}$$

If $\beta > 0$, then can improve cost with $\pi(1) = 2$ and $\pi(2) = 1$.

Least squares problem ($d = 1$)

Given $(\boldsymbol{x}_i)_{i=1}^n$ and $(y_i)_{i=1}^n$ from $\mathbb{R}$, minimize

$$F(\beta, \pi) \;:=\; \sum_{i=1}^{n} \left(x_i \beta - y_{\pi(i)} \right)^2 .$$

$$\begin{array}{c} (3\beta - \boxed{2})^2 \\ (4\beta - \boxed{1})^2 \\ \vdots \\ (6\beta - 7)^2 \end{array} \;>\; \begin{array}{c} (3\beta - \boxed{1})^2 \\ (4\beta - \boxed{2})^2 \\ \vdots \\ (6\beta - 7)^2 \end{array}$$

If $\beta > 0$, then can improve cost with $\pi(1) = 2$ and $\pi(2) = 1$.

$$25\beta^2 - 20\beta + 5 + \cdots \;>\; 25\beta^2 - 22\beta + 5 + \cdots$$

Algorithm for least squares problem ($d = 1$) [PWC'16]

1. "Guess" sign of optimal β. (Only two possibilities.)

Algorithm for least squares problem ($d = 1$) [PWC'16]

1. "Guess" sign of optimal β. (Only two possibilities.)
2. Assuming WLOG that $x_1\beta \leq x_2\beta \cdots \leq x_n\beta$,
   find optimal π such that $y_{\pi(1)} \leq y_{\pi(2)} \leq \cdots \leq y_{\pi(n)}$
   (via sorting).

Algorithm for least squares problem ($d = 1$) [PWC'16]

1. "Guess" sign of optimal β. (Only two possibilities.)
2. Assuming WLOG that $x_1\beta \leq x_2\beta \cdots \leq x_n\beta$, find optimal π such that $y_{\pi(1)} \leq y_{\pi(2)} \leq \cdots \leq y_{\pi(n)}$ (via sorting).
3. Solve classical least squares problem

$$\min_{\beta \in \mathbb{R}} \sum_{i=1}^{n} (x_i\beta - y_{\pi(i)})^2$$

to get optimal β.

Algorithm for least squares problem ($d = 1$) [PWC'16]

1. "Guess" sign of optimal β. (Only two possibilities.)
2. Assuming WLOG that $x_1\beta \le x_2\beta \cdots \le x_n\beta$, find optimal π such that $y_{\pi(1)} \le y_{\pi(2)} \le \cdots \le y_{\pi(n)}$ (via sorting).
3. Solve classical least squares problem

$$\min_{\beta \in \mathbb{R}} \sum_{i=1}^{n} (x_i\beta - y_{\pi(i)})^2$$

   to get optimal β.

Overall running time: $O(n \log n)$.

Algorithm for least squares problem ($d = 1$) [PWC'16]

1. "Guess" sign of optimal β. (Only two possibilities.)
2. Assuming WLOG that $x_1\beta \leq x_2\beta \cdots \leq x_n\beta$, find optimal π such that $y_{\pi(1)} \leq y_{\pi(2)} \leq \cdots \leq y_{\pi(n)}$ (via sorting).
3. Solve classical least squares problem

$$\min_{\beta \in \mathbb{R}} \sum_{i=1}^{n} (x_i\beta - y_{\pi(i)})^2$$

to get optimal β.

Overall running time: $O(n \log n)$.

What about $d > 1$?

Alternating minimization

Pick initial $\hat{\boldsymbol{\beta}} \in \mathbb{R}^d$ (e.g., randomly).

Loop until convergence:

$$\hat{\pi} \leftarrow \operatorname*{arg\,min}_{\pi \in S_n} \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}} - y_{\pi(i)} \right)^2 .$$

$$\hat{\boldsymbol{\beta}} \leftarrow \operatorname*{arg\,min}_{\boldsymbol{\beta} \in \mathbb{R}^d} \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{\beta} - y_{\hat{\pi}(i)} \right)^2 .$$

Alternating minimization

Pick initial $\hat{\boldsymbol{\beta}} \in \mathbb{R}^d$ (e.g., randomly).

Loop until convergence:

$$\hat{\pi} \leftarrow \operatorname*{arg\,min}_{\pi \in S_n} \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}} - y_{\pi(i)} \right)^2 .$$

$$\hat{\boldsymbol{\beta}} \leftarrow \operatorname*{arg\,min}_{\boldsymbol{\beta} \in \mathbb{R}^d} \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{\beta} - y_{\hat{\pi}(i)} \right)^2 .$$

- Each loop-iteration efficiently computable.

Alternating minimization

Pick initial $\hat{\boldsymbol{\beta}} \in \mathbb{R}^d$ (e.g., randomly).

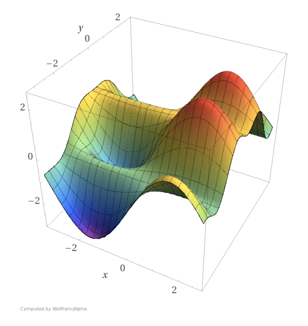
Loop until convergence:

$$\hat{\pi} \;\leftarrow\; \operatorname*{arg\,min}_{\pi \in S_n} \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}} - y_{\pi(i)} \right)^2 .$$

$$\hat{\boldsymbol{\beta}} \;\leftarrow\; \operatorname*{arg\,min}_{\boldsymbol{\beta} \in \mathbb{R}^d} \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{\beta} - y_{\hat{\pi}(i)} \right)^2 .$$

- Each loop-iteration efficiently computable.
- But can get stuck in local minima.

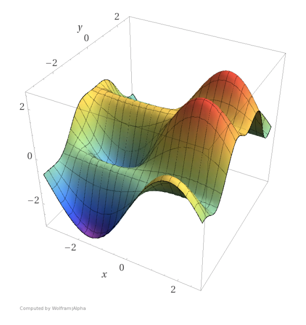
Alternating minimization



(Image credit: `Wolfram|Alpha`)

- Each loop-iteration efficiently computable.
- But can get stuck in local minima.

Alternating minimization



(Image credit: `Wolfram|Alpha`)

- Each loop-iteration efficiently computable.
- But can get stuck in local minima. So try many initial $\hat{\beta} \in \mathbb{R}^d$.
  (**Open**: How many restarts? How many iterations?)

Approximation result

Theorem (H., Shi, & Sun, 2017)
There is an algorithm that given any inputs $(\boldsymbol{x}_i)_{i=1}^n$, $(y_i)_{i=1}^n$, *and* $\epsilon \in (0,1)$, *returns a* $(1+\epsilon)$*-approximate solution to the least squares problem in time*

$$\left(\frac{n}{\epsilon}\right)^{O(d)} + \mathrm{poly}(n,d)\,.$$

Approximation result

Theorem (H., Shi, & Sun, 2017)
There is an algorithm that given any inputs $(\boldsymbol{x}_i)_{i=1}^n$, $(y_i)_{i=1}^n$, *and* $\epsilon \in (0,1)$, *returns a* $(1+\epsilon)$*-approximate solution to the least squares problem in time*

$$\left(\frac{n}{\epsilon}\right)^{O(d)} + \mathrm{poly}(n,d)\,.$$

Recall: Brute-force solution needs $\Omega(n!)$ time.

(No other previous algorithm with approximation guarantee.)

Statistical recovery of β^*: algorithms and lower-bounds

Motivation

When does the best fit model shed light on the "truth" (π^* & $\boldsymbol{\beta}^*$)?

Motivation

When does the best fit model shed light on the “truth” (π^* & $\boldsymbol{\beta}^*$)?

Approach: Study question in context of statistical model for data.

Motivation

When does the best fit model shed light on the "truth" (π^* & $\boldsymbol{\beta}^*$)?

Approach: Study question in context of statistical model for data.

1. Understand information-theoretic limits on recovering truth.
2. Natural "average-case" setting for algorithms.

Statistical model

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_{\pi^*(1)}^\top \\ \boldsymbol{x}_{\pi^*(2)}^\top \\ \vdots \\ \boldsymbol{x}_{\pi^*(n)}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}^* \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

Assume $(\boldsymbol{x}_i)_{i=1}^n$ iid from $\mathbb{P}$ and $(\varepsilon_i)_{i=1}^n$ iid from $\mathrm{N}(0, \sigma^2)$.

Statistical model

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_{\pi^*(1)}^\top \\ \boldsymbol{x}_{\pi^*(2)}^\top \\ \vdots \\ \boldsymbol{x}_{\pi^*(n)}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}^* \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

Assume $(\boldsymbol{x}_i)_{i=1}^n$ iid from $\mathbb{P}$ and $(\varepsilon_i)_{i=1}^n$ iid from $\mathrm{N}(0, \sigma^2)$.

Recoverability of $\boldsymbol{\beta}^*$ depends on **signal-to-noise ratio**:

$$\mathsf{SNR} := \frac{\|\boldsymbol{\beta}^*\|_2^2}{\sigma^2}.$$

Statistical model

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_{\pi^*(1)}^\top \\ \boldsymbol{x}_{\pi^*(2)}^\top \\ \vdots \\ \boldsymbol{x}_{\pi^*(n)}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}^* \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

Assume $(\boldsymbol{x}_i)_{i=1}^n$ iid from $\mathbb{P}$ and $(\varepsilon_i)_{i=1}^n$ iid from $\mathrm{N}(0, \sigma^2)$.

Recoverability of $\boldsymbol{\beta}^*$ depends on **signal-to-noise ratio**:

$$\mathrm{SNR} \;:=\; \frac{\|\boldsymbol{\beta}^*\|_2^2}{\sigma^2} .$$

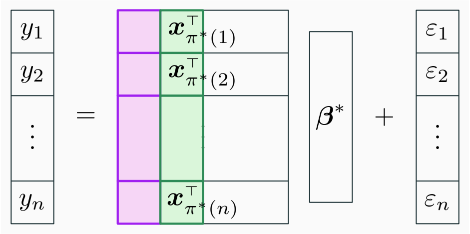
Classical setting (where π^* is known):
Just need $\mathrm{SNR} \gtrsim d/n$ to approximately recover $\boldsymbol{\beta}^*$.

High-level intuition

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_{\pi^*(1)}^\top \\ \boldsymbol{x}_{\pi^*(2)}^\top \\ \vdots \\ \boldsymbol{x}_{\pi^*(n)}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}^* \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

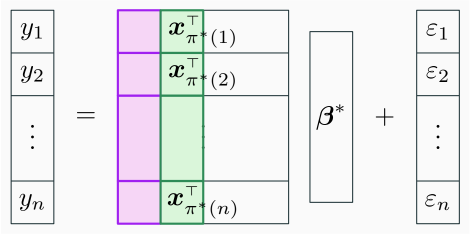
High-level intuition

Suppose β^* is either $\boldsymbol{e}_1 = (1, 0, 0, \ldots, 0)$ or $\boldsymbol{e}_2 = (0, 1, 0, \ldots, 0)$.



High-level intuition

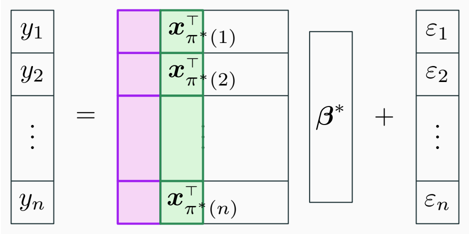
Suppose β^* is either $e_1 = (1, 0, 0, \ldots, 0)$ or $e_2 = (0, 1, 0, \ldots, 0)$.



π^* **known**: distinguishability of e_1 and e_2 *can improve with* n.

High-level intuition

Suppose $\boldsymbol{\beta}^*$ is either $\boldsymbol{e}_1 = (1, 0, 0, \ldots, 0)$ or $\boldsymbol{e}_2 = (0, 1, 0, \ldots, 0)$.



π^* **known**: distinguishability of $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ *can improve with* n.

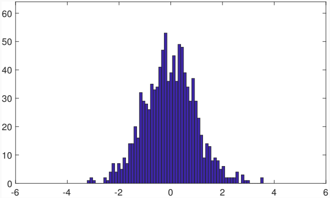
π^* ***unknown***: distinguishability is less clear.

$$\lbag y_i \rbag_{i=1}^n = \begin{cases} \lbag x_{i,1} \rbag_{i=1}^n + \mathrm{N}(0, \sigma^2) & \text{if } \boldsymbol{\beta}^* = \boldsymbol{e}_1, \\ \lbag x_{i,2} \rbag_{i=1}^n + \mathrm{N}(0, \sigma^2) & \text{if } \boldsymbol{\beta}^* = \boldsymbol{e}_2. \end{cases}$$

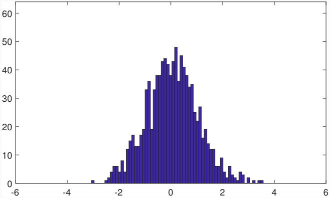
($\lbag \cdot \rbag$ denotes *unordered multi-set*.)

Effect of noise

Without noise ($\mathbb{P} = \mathrm{N}(0, \boldsymbol{I}_d)$)



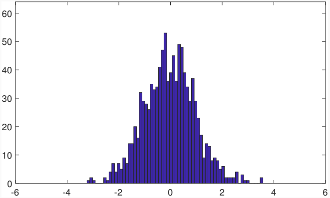
$\{x_{i,1}\}_{i=1}^{n}$



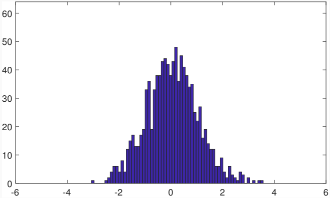
$\{x_{i,2}\}_{i=1}^{n}$

Effect of noise

Without noise ($\mathbb{P} = \mathrm{N}(0, \boldsymbol{I}_d)$)

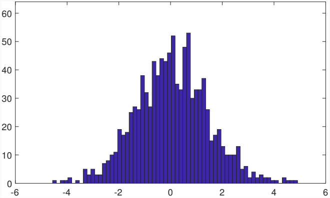


$\{x_{i,1}\}_{i=1}^n$



$\{x_{i,2}\}_{i=1}^n$

With noise



$??? + \mathrm{N}(0, \sigma^2)$

Lower bound on SNR

Theorem (H., Shi, & Sun, 2017)

For $\mathbb{P} = \mathrm{N}(0, \boldsymbol{I}_d)$, *no estimator* $\hat{\boldsymbol{\beta}}$ *can guarantee*

$$\mathbb{E}\left[\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2\right] \;\leq\; \frac{\|\boldsymbol{\beta}^*\|_2}{3}$$

unless

$$\mathsf{SNR} \;\geq\; C \cdot \frac{d}{\log\log(n)} \, .$$

Lower bound on SNR

Theorem (H., Shi, & Sun, 2017)

For $\mathbb{P} = \mathrm{N}(0, \boldsymbol{I}_d)$, *no estimator* $\hat{\boldsymbol{\beta}}$ *can guarantee*

$$\mathbb{E}\left[\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2\right] \;\leq\; \frac{\|\boldsymbol{\beta}^*\|_2}{3}$$

unless

$$\mathsf{SNR} \;\geq\; C \cdot \frac{d}{\log\log(n)} \,.$$

"Known correspondence" setting: SNR $\gtrsim d/n$ suffices.

Lower bound on SNR

Theorem (H., Shi, & Sun, 2017)

For $\mathbb{P} = \mathrm{N}(0, \boldsymbol{I}_d)$, *no estimator* $\hat{\beta}$ *can guarantee*

$$\mathbb{E}\left[\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2\right] \;\leq\; \frac{\|\boldsymbol{\beta}^*\|_2}{3}$$

unless

$$\mathsf{SNR} \;\geq\; C \cdot \frac{d}{\log\log(n)} \,.$$

"Known correspondence" setting: SNR $\gtrsim d/n$ suffices.

Another theorem: for $\mathbb{P} = \mathrm{Uniform}([-1,1]^d)$, must have SNR $\geq 1/9$, even as $n \to \infty$.

High SNR regime

High SNR regime

Previous works (Unnikrishnan, Haghighatshoar, & Vetterli, 2015; Pananjady, Wainwright, & Courtade, 2016):

If SNR $\gg \text{poly}(n)$, then can recover π^* (and $\boldsymbol{\beta}^*$, approximately) using Maximum Likelihood Estimation, i.e., least squares.

High SNR **regime**

Previous works (Unnikrishnan, Haghighatshoar, & Vetterli, 2015; Pananjady, Wainwright, & Courtade, 2016):

If SNR $\gg \mathrm{poly}(n)$, then can recover π^* (and $\boldsymbol{\beta}^*$, approximately) using Maximum Likelihood Estimation, i.e., least squares.

Related ($d = 1$): broken random sample (DeGroot and Goel, 1980). Estimate sign of correlation between x_i and y_i.

Have estimator for $\mathrm{sign}(\beta^*)$ that is correct w.p. $1 - \tilde{O}(\mathsf{SNR}^{-1/4})$.

High SNR regime

Previous works (Unnikrishnan, Haghighatshoar, & Vetterli, 2015; Pananjady, Wainwright, & Courtade, 2016):

If SNR $\gg \mathrm{poly}(n)$, then can recover π^* (and $\boldsymbol{\beta}^*$, approximately) using Maximum Likelihood Estimation, i.e., least squares.

Related ($d = 1$): broken random sample (DeGroot and Goel, 1980). Estimate sign of correlation between x_i and y_i.

Have estimator for $\mathrm{sign}(\beta^*)$ that is correct w.p. $1 - \tilde{O}(\mathrm{SNR}^{-1/4})$.

Does high SNR also permit efficient algorithms?

(Recall: our approximate MLE algorithm has running time $n^{O(d)}$.)

Average-case recovery with very high SNR

Noise-free setting (SNR $= \infty$)

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_{\pi^*(0)}^\top \\ \boldsymbol{x}_{\pi^*(1)}^\top \\ \vdots \\ \boldsymbol{x}_{\pi^*(n)}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}^* \end{bmatrix}$$

Assume $(\boldsymbol{x}_i)_{i=0}^n$ iid from $\mathrm{N}(0, \boldsymbol{I}_d)$.

Noise-free setting (SNR $= \infty$)

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_0^\top \\ \boldsymbol{x}_{\pi^*(1)}^\top \\ \vdots \\ \boldsymbol{x}_{\pi^*(n)}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}^* \end{bmatrix}$$

Assume $(\boldsymbol{x}_i)_{i=0}^n$ iid from $\mathrm{N}(0, \boldsymbol{I}_d)$.

Also assume $\pi^*(0) = 0$.

Noise-free setting (SNR $= \infty$)

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_0^\top \\ \boldsymbol{x}_{\pi^*(1)}^\top \\ \vdots \\ \boldsymbol{x}_{\pi^*(n)}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}^* \end{bmatrix}$$

Assume $(\boldsymbol{x}_i)_{i=0}^n$ iid from $\mathrm{N}(0, \boldsymbol{I}_d)$.

Also assume $\pi^*(0) = 0$.

If $n + 1 \geq d$, then recovery of π^* gives exact recovery of β^* (a.s.).

Noise-free setting (SNR $= \infty$)

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_0^\top \\ \boldsymbol{x}_{\pi^*(1)}^\top \\ \vdots \\ \boldsymbol{x}_{\pi^*(n)}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}^* \end{bmatrix}$$

Assume $(\boldsymbol{x}_i)_{i=0}^n$ iid from $\mathrm{N}(0, \boldsymbol{I}_d)$.

Also assume $\pi^*(0) = 0$.

If $n + 1 \geq d$, then recovery of π^* gives exact recovery of β^* (a.s.).

We'll assume $n + 1 \geq d + 1$ (i.e., $n \geq d$).

Noise-free setting (SNR $= \infty$)

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_0^\top \\ \boldsymbol{x}_{\pi^*(1)}^\top \\ \vdots \\ \boldsymbol{x}_{\pi^*(n)}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}^* \end{bmatrix}$$

Assume $(\boldsymbol{x}_i)_{i=0}^n$ iid from $\mathrm{N}(0, \boldsymbol{I}_d)$.

Also assume $\pi^*(0) = 0$.

If $n + 1 \geq d$, then recovery of π^* gives exact recovery of β^* (a.s.).

We'll assume $n + 1 \geq d + 1$ (i.e., $n \geq d$).

Claim: $n \geq d$ suffices to recover π^* with high probability.

Result on exact recovery

Theorem (H., Shi, & Sun, 2017)
In the noise-free setting, there is a $\mathrm{poly}(n, d)$*-time*⋆ *algorithm that returns* π^* *and* β^* *with high probability.*

⋆Assuming problem is appropriately discretized.

Main idea: hidden subset

Measurements:

$$y_0 \;=\; \boldsymbol{x}_0^\top \boldsymbol{\beta}^*\,; \qquad\qquad y_i \;=\; \boldsymbol{x}_{\pi^*(i)}^\top \boldsymbol{\beta}^*\,, \quad i = 1, \ldots, n\,.$$

Main idea: hidden subset

Measurements:

$$y_0 \;=\; \boldsymbol{x}_0^\top \boldsymbol{\beta}^* \,; \qquad\qquad y_i \;=\; \boldsymbol{x}_{\pi^*(i)}^\top \boldsymbol{\beta}^* \,, \quad i = 1, \dots, n \,.$$

For simplicity: assume $n = d$, and $\boldsymbol{x}_i = \boldsymbol{e}_i$ for $i = 1, \dots, d$, so

$$\wr y_1, \dots, y_d \wr \;=\; \wr \beta_1^*, \dots, \beta_d^* \wr \,.$$

Main idea: hidden subset

Measurements:

$$y_0 \;=\; \boldsymbol{x}_0^\top \boldsymbol{\beta}^* \,; \qquad\qquad y_i \;=\; \boldsymbol{x}_{\pi^*(i)}^\top \boldsymbol{\beta}^* \,, \quad i = 1, \dots, n \,.$$

For simplicity: assume $n = d$, and $\boldsymbol{x}_i = \boldsymbol{e}_i$ for $i = 1, \dots, d$, so

$$\wr y_1, \dots, y_d \int \;=\; \wr \beta_1^*, \dots, \beta_d^* \int \,.$$

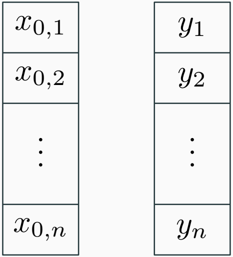
We also know:

$$y_0 \;=\; \boldsymbol{x}_0^\top \boldsymbol{\beta}^* \;=\; \sum_{j=1}^{d} x_{0,j} \beta_j^* \,.$$

Reduction to Subset Sum

$$y_0 = \boldsymbol{x}_0^\top \boldsymbol{\beta}^* = \sum_{j=1}^{d} x_{0,j} \beta_j^*$$

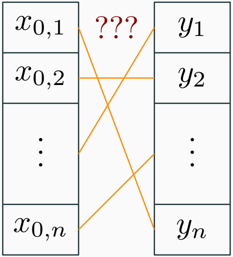
$$= \sum_{i=1}^{d} \sum_{j=1}^{d} x_{0,j} y_i \cdot 1\{\pi^*(i) = j\}$$



Reduction to Subset Sum

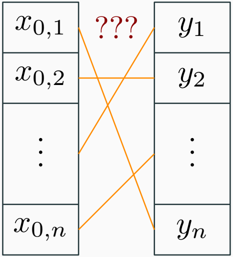
$$y_0 = \boldsymbol{x}_0^\top \boldsymbol{\beta}^* = \sum_{j=1}^{d} x_{0,j}\beta_j^*$$

$$= \sum_{i=1}^{d}\sum_{j=1}^{d} x_{0,j} y_i \cdot 1\{\pi^*(i) = j\}$$



Reduction to Subset Sum

$$y_0 = \boldsymbol{x}_0^\top \boldsymbol{\beta}^* = \sum_{j=1}^{d} x_{0,j}\beta_j^*$$
$$= \sum_{i=1}^{d}\sum_{j=1}^{d} x_{0,j} y_i \cdot 1\{\pi^*(i) = j\}$$

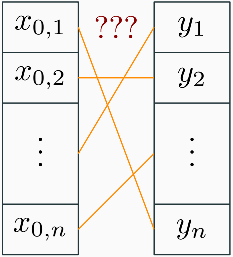


- d^2 "source" numbers $c_{i,j} := x_{0,j} y_i$, "target" sum y_0.

  The subset $\{c_{i,j} : \pi^*(i) = j\}$ adds up to y_0.

Reduction to Subset Sum

$$y_0 = \boldsymbol{x}_0^\top \boldsymbol{\beta}^* = \sum_{j=1}^{d} x_{0,j}\beta_j^*$$
$$= \sum_{i=1}^{d}\sum_{j=1}^{d} x_{0,j} y_i \cdot 1\{\pi^*(i) = j\}$$



- d^2 "source" numbers $c_{i,j} := x_{0,j} y_i$, "target" sum y_0.
  The subset $\{c_{i,j} : \pi^*(i) = j\}$ adds up to y_0.

Subset Sum problem.

NP-Completeness of Subset Sum (a.k.a. "Knapsack")

REDUCIBILITY AMONG COMBINATORIAL PROBLEMS†

Richard M. Karp

University of California at Berkeley

18. KNAPSACK

INPUT: $(a_1, a_2, \ldots, a_r, b) \in Z^{n+1}$

PROPERTY: $\Sigma\, a_j x_j = b$ has a 0-1 solution.

(Karp, 1972)

Easiness of Subset Sum

- But Subset Sum is only "weakly" NP-hard
  (efficient algorithm exists for unary-encoded inputs).

Easiness of Subset Sum

- But Subset Sum is only “weakly” NP-hard (efficient algorithm exists for unary-encoded inputs).
- **Lagarias & Odlyzko (1983)**: solving certain random instances can be reduced to solving Approximate Shortest Vector Problem in lattices.

Easiness of Subset Sum

- But Subset Sum is only “weakly” NP-hard (efficient algorithm exists for unary-encoded inputs).
- **Lagarias & Odlyzko (1983)**: solving certain random instances can be reduced to solving Approximate Shortest Vector Problem in lattices.
- **Lenstra, Lenstra, & Lovász (1982)**: efficient algorithm to solve Approximate SVP.

Easiness of Subset Sum

- But Subset Sum is only “weakly” NP-hard
  (efficient algorithm exists for unary-encoded inputs).
- **Lagarias & Odlyzko (1983)**: solving certain random instances can be reduced to solving Approximate Shortest Vector Problem in lattices.
- **Lenstra, Lenstra, & Lovász (1982)**: efficient algorithm to solve Approximate SVP.
- Our algorithm is based on similar reduction but requires a somewhat different analysis.

Reducing subset sum to shortest vector problem

Lagarias & Odlyzko (1983): random instances of Subset Sum *efficiently solvable* when N source numbers $c_1, \ldots, c_N$ chosen independently and u.a.r. from sufficiently wide interval of $\mathbb{Z}$.

Reducing subset sum to shortest vector problem

Lagarias & Odlyzko (1983): random instances of Subset Sum *efficiently solvable* when N source numbers $c_1, \ldots, c_N$ chosen independently and u.a.r. from sufficiently wide interval of $\mathbb{Z}$.

Main idea: (w.h.p.) every incorrect subset will "miss" the target sum T by noticeable amount.

Reducing subset sum to shortest vector problem

Lagarias & Odlyzko (1983): random instances of Subset Sum *efficiently solvable* when N source numbers $c_1, \ldots, c_N$ chosen independently and u.a.r. from sufficiently wide interval of $\mathbb{Z}$.

Main idea: (w.h.p.) every incorrect subset will "miss" the target sum T by noticeable amount.

Reduction: construct lattice basis in $\mathbb{R}^{N+1}$ such that

- correct subset of basis vectors gives short lattice vector $\boldsymbol{v}_\star$;
- any other lattice vector $\not\propto \boldsymbol{v}_\star$ is more than $2^{N/2}$-times longer.

$$\left[\begin{array}{c|c|c|c} \boldsymbol{b}_0 & \boldsymbol{b}_1 & \cdots & \boldsymbol{b}_N \end{array}\right] := \left[\begin{array}{c|ccc} 0 & & \boldsymbol{I}_N & \\ \hline MT & -Mc_1 & \cdots & -Mc_N \end{array}\right]$$

for sufficiently large $M > 0$.

Our random subset sum instance

Catch: Our source numbers $c_{i,j} = y_i \boldsymbol{x}_j^\top \boldsymbol{x}_0$ are **not independent**, and **not uniformly distributed** on some wide interval of $\mathbb{Z}$.

Our random subset sum instance

Catch: Our source numbers $c_{i,j} = y_i \boldsymbol{x}_j^\top \boldsymbol{x}_0$ are **not independent**, and **not uniformly distributed** on some wide interval of $\mathbb{Z}$.

- Instead, have some joint density derived from $\mathrm{N}(0, 1)$.

Our random subset sum instance

Catch: Our source numbers $c_{i,j} = y_i \boldsymbol{x}_j^\top \boldsymbol{x}_0$ are **not independent**, and **not uniformly distributed** on some wide interval of $\mathbb{Z}$.

- Instead, have some joint density derived from $\mathrm{N}(0, 1)$.
- To show that Lagarias & Odlyzko reduction still works, use Gaussian anti-concentration for quadratic and quartic forms.

Our random subset sum instance

Catch: Our source numbers $c_{i,j} = y_i \boldsymbol{x}_j^\top \boldsymbol{x}_0$ are **not independent**, and **not uniformly distributed** on some wide interval of $\mathbb{Z}$.

- Instead, have some joint density derived from $\mathrm{N}(0,1)$.
- To show that Lagarias & Odlyzko reduction still works, use Gaussian anti-concentration for quadratic and quartic forms.

  Key lemma: (w.h.p.) for every $\boldsymbol{Z} \in \mathbb{Z}^{d \times d}$ that is not an integer multiple of permutation matrix corresponding to π^*,

$$\left| y_0 - \sum_{i,j} Z_{i,j} \cdot c_{i,j} \right| \geq \frac{1}{2^{\mathrm{poly}(d)}} \cdot \|\boldsymbol{\beta}^*\|_2 .$$

Some remarks

- In general, $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ are not $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_d$, but similar reduction works via Moore-Penrose pseudoinverse.

Some remarks

- In general, $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ are not $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_d$, but similar reduction works via Moore-Penrose pseudoinverse.
- Algorithm strongly exploits assumption of noise-free measurements. **Unlikely to tolerate much noise**.

  Open problem:
  robust efficient algorithm in high SNR setting.

Correspondence retrieval

Correspondence retrieval problem

Goal: recover k unknown "signals" $\boldsymbol{\beta}_1^*, \ldots, \boldsymbol{\beta}_k^* \in \mathbb{R}^d$.

Correspondence retrieval problem

Goal: recover k unknown "signals" $\boldsymbol{\beta}_1^*, \ldots, \boldsymbol{\beta}_k^* \in \mathbb{R}^d$.

Measurements: $(\boldsymbol{x}_i, \mathcal{Y}_i)$ for $i = 1, \ldots, n$, where

- $(\boldsymbol{x}_i)$ iid from $\mathrm{N}(0, \boldsymbol{I}_d)$;
- $\mathcal{Y}_i = \wr \boldsymbol{x}_i^\top \boldsymbol{\beta}_1^* + \varepsilon_{i,1}, \ldots, \boldsymbol{x}_i^\top \boldsymbol{\beta}_k^* + \varepsilon_{i,k} \int$ as unordered multi-set;
- $(\varepsilon_{i,j})$ iid from $\mathrm{N}(0, \sigma^2)$.

Correspondence across measurements is lost.

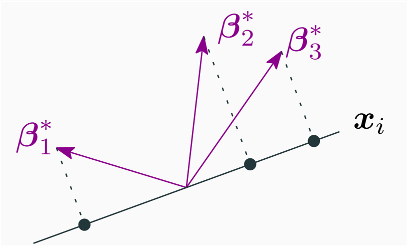
Correspondence retrieval problem

Goal: recover k unknown "signals" $\boldsymbol{\beta}_1^*, \ldots, \boldsymbol{\beta}_k^* \in \mathbb{R}^d$.

Measurements: $(\boldsymbol{x}_i, \mathcal{Y}_i)$ for $i = 1, \ldots, n$, where

- $(\boldsymbol{x}_i)$ iid from $\mathrm{N}(0, \boldsymbol{I}_d)$;
- $\mathcal{Y}_i = \wr \boldsymbol{x}_i^\top \boldsymbol{\beta}_1^* + \varepsilon_{i,1}, \ldots, \boldsymbol{x}_i^\top \boldsymbol{\beta}_k^* + \varepsilon_{i,k} \int$ as unordered multi-set;
- $(\varepsilon_{i,j})$ iid from $\mathrm{N}(0, \sigma^2)$.

Correspondence across measurements is lost.



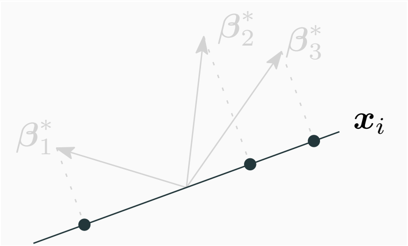
Correspondence retrieval problem

Goal: recover k unknown "signals" $\boldsymbol{\beta}_1^*, \dots, \boldsymbol{\beta}_k^* \in \mathbb{R}^d$.

Measurements: $(\boldsymbol{x}_i, \mathcal{Y}_i)$ for $i = 1, \dots, n$, where

- $(\boldsymbol{x}_i)$ iid from $\mathrm{N}(0, \boldsymbol{I}_d)$;
- $\mathcal{Y}_i = \wr \boldsymbol{x}_i^\top \boldsymbol{\beta}_1^* + \varepsilon_{i,1}, \dots, \boldsymbol{x}_i^\top \boldsymbol{\beta}_k^* + \varepsilon_{i,k} \int$ as unordered multi-set;
- $(\varepsilon_{i,j})$ iid from $\mathrm{N}(0, \sigma^2)$.

Correspondence across measurements is lost.



Special cases

- $k = 1$: classical linear regression regression model.

Special cases

- $k = 1$: classical linear regression regression model.
- $k = 2$ and $\boldsymbol{\beta}_1^* = -\boldsymbol{\beta}_2^*$: (real variant of) *phase retrieval*.

  Note that $\wr \boldsymbol{x}_i^\top \boldsymbol{\beta}^*, -\boldsymbol{x}_i^\top \boldsymbol{\beta}^* \int$ has same information as $|\boldsymbol{x}_i^\top \boldsymbol{\beta}^*|$.

  Existing methods require $n > 2d$.

Algorithmic results (Andoni, H., Shi, & Sun, 2017)

- **Noise-free setting** (i.e., $\sigma = 0$):
  Algorithm based on reduction to Subset Sum that requires $n \geq d + 1$, which is optimal.

Algorithmic results (Andoni, H., Shi, & Sun, 2017)

- **Noise-free setting** (i.e., $\sigma = 0$):
  Algorithm based on reduction to Subset Sum that requires $n \geq d + 1$, which is optimal.
- **General setting**:
  Method-of-moments algorithm that requires $n \geq d \cdot \mathrm{poly}(k)$.

Algorithmic results (Andoni, H., Shi, & Sun, 2017)

- **Noise-free setting** (i.e., $\sigma = 0$):
  Algorithm based on reduction to Subset Sum that requires $n \geq d + 1$, which is optimal.
- **General setting**:
  Method-of-moments algorithm that requires $n \geq d \cdot \mathrm{poly}(k)$.
  I.e., based on forming averages over the data, like:

$$\frac{1}{n}\sum_{i=1}^{n}\left(\sum_{y_j \in \mathcal{Y}_i} y_j^2\right)\boldsymbol{x}_i\boldsymbol{x}_i^\top .$$

Algorithmic results (Andoni, H., Shi, & Sun, 2017)

- **Noise-free setting** (i.e., $\sigma = 0$):
  Algorithm based on reduction to Subset Sum that requires $n \geq d + 1$, which is optimal.
- **General setting**:
  Method-of-moments algorithm that requires $n \geq d \cdot \text{poly}(k)$.
  I.e., based on forming averages over the data, like:

$$\frac{1}{n} \sum_{i=1}^{n} \left(\sum_{y_j \in \mathcal{Y}_i} y_j^2 \right) \boldsymbol{x}_i \boldsymbol{x}_i^\top .$$

Questions: SNR limits? Sub-optimality of "method-of-moments"?

Closing remarks and open problems

Closing remarks and open problems

Learning without correspondence is challenging for computation and statistics.

Closing remarks and open problems

Learning without correspondence is challenging for computation and statistics.

- **Computational and information-theoretic hardness** show striking contrast to “known correspondence” settings.

Closing remarks and open problems

Learning without correspondence is challenging for computation and statistics.

- **Computational and information-theoretic hardness** show striking contrast to "known correspondence" settings.
- **New (and unexpected?) algorithmic techniques** in worst-case and average-case settings.

Closing remarks and open problems

Learning without correspondence is challenging for computation and statistics.

- **Computational and information-theoretic hardness** show striking contrast to "known correspondence" settings.
- **New (and unexpected?) algorithmic techniques** in worst-case and average-case settings.
- **Open problems**:
  Close gap between SNR lower and upper bounds?
  Lower bounds for correspondence retrieval?
  Faster/more robust algorithms?
  (Smoothed) analysis of alternating minimization?

Acknowledgements

Collaborators: Alexandr Andoni (Columbia), Kevin Shi (Columbia), Xiaorui Sun (Microsoft Research).

Funding: NSF (DMR-1534910, IIS-1563785), Sloan Research Fellowship, Bloomberg Data Science Research Grant.

Hospitality: Simons Institute for the Theory of Computing (UCB).

Thank you

Beating brute-force search: "realizable" case

"Realizable" case: Suppose there exist $\boldsymbol{\beta}_\star \in \mathbb{R}^d$ and $\pi_\star \in S_n$ s.t.

$$y_{\pi_\star(i)} \; = \; \boldsymbol{x}_i^\top \boldsymbol{\beta}_\star \,, \quad i \in [n] \,.$$

Beating brute-force search: "realizable" case

"Realizable" case: Suppose there exist $\boldsymbol{\beta}_\star \in \mathbb{R}^d$ and $\pi_\star \in S_n$ s.t.

$$y_{\pi_\star(i)} = \boldsymbol{x}_i^\top \boldsymbol{\beta}_\star , \quad i \in [n] .$$

Solution is determined by action of $\pi_\star$ on d points

(assume $\dim(\mathrm{span}(\boldsymbol{x}_i)_{i=1}^d) = d$).

Beating brute-force search: "realizable" case

"Realizable" case: Suppose there exist $\boldsymbol{\beta}_\star \in \mathbb{R}^d$ and $\pi_\star \in S_n$ s.t.

$$y_{\pi_\star(i)} \;=\; \boldsymbol{x}_i^\top \boldsymbol{\beta}_\star \,, \quad i \in [n] \,.$$

Solution is determined by action of $\pi_\star$ on d points

(assume $\dim(\operatorname{span}(\boldsymbol{x}_i)_{i=1}^d) = d$).

Algorithm:

- Find subset of d linearly independent points $\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \ldots, \boldsymbol{x}_{i_d}$.
- "Guess" values of $\pi_\star(i_j) \in [d]$, $j \in [d]$.
- Solve linear system $y_{\pi_\star(i_j)} = \boldsymbol{x}_{i_j}^\top \boldsymbol{\beta}$, $j \in [d]$, for $\boldsymbol{\beta} \in \mathbb{R}^d$.
- To check correctness of $\hat{\boldsymbol{\beta}}$: compute $\hat{y}_i := \boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}$, $i \in [n]$, and check if $\min_{\pi \in S_n} \sum_{i=1}^n \left(y_{\pi(i)} - \hat{y}_i\right)^2 = 0$.

Beating brute-force search: "realizable" case

"Realizable" case: Suppose there exist $\boldsymbol{\beta}_\star \in \mathbb{R}^d$ and $\pi_\star \in S_n$ s.t.

$$y_{\pi_\star(i)} \;=\; \boldsymbol{x}_i^\top \boldsymbol{\beta}_\star\,, \quad i \in [n]\,.$$

Solution is determined by action of $\pi_\star$ on d points
(assume $\dim(\operatorname{span}(\boldsymbol{x}_i)_{i=1}^d) = d$).

Algorithm:

- Find subset of d linearly independent points $\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \ldots, \boldsymbol{x}_{i_d}$.
- "Guess" values of $\pi_\star(i_j) \in [d]$, $j \in [d]$.
- Solve linear system $y_{\pi_\star(i_j)} = \boldsymbol{x}_{i_j}^\top \boldsymbol{\beta}$, $j \in [d]$, for $\boldsymbol{\beta} \in \mathbb{R}^d$.
- To check correctness of $\hat{\boldsymbol{\beta}}$: compute $\hat{y}_i := \boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}$, $i \in [n]$, and check if $\min_{\pi \in S_n} \sum_{i=1}^n \left(y_{\pi(i)} - \hat{y}_i\right)^2 = 0$.

"Guess" means "enumerate over $\binom{n}{d}$ choices"; rest is $\operatorname{poly}(n, d)$.

Beating brute-force search: general case

General case: solution may not be determined by only d points.

Beating brute-force search: general case

General case: solution may not be determined by only d points.

But, for any RHS $\boldsymbol{b} \in \mathbb{R}^n$, there exist $\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \ldots, \boldsymbol{x}_{i_d}$ s.t. every $\hat{\boldsymbol{\beta}} \in \arg\min_{\boldsymbol{\beta} \in \mathbb{R}^d} \sum_{j=1}^{d} \left(\boldsymbol{x}_{i_j}^\top \boldsymbol{\beta} - b_{i_j}\right)^2$ satisfies

$$\sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}} - b_i\right)^2 \;\le\; (d+1) \cdot \min_{\boldsymbol{\beta} \in \mathbb{R}^d} \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{\beta} - b_i\right)^2 \;.$$

Beating brute-force search: general case

General case: solution may not be determined by only d points.

But, for any RHS $\boldsymbol{b} \in \mathbb{R}^n$, there exist $\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \ldots, \boldsymbol{x}_{i_d}$ s.t. every $\hat{\boldsymbol{\beta}} \in \arg\min_{\boldsymbol{\beta} \in \mathbb{R}^d} \sum_{j=1}^{d} (\boldsymbol{x}_{i_j}^\top \boldsymbol{\beta} - b_{i_j})^2$ satisfies

$$\sum_{i=1}^{n} (\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}} - b_i)^2 \ \leq \ (d+1) \cdot \min_{\boldsymbol{\beta} \in \mathbb{R}^d} \sum_{i=1}^{n} (\boldsymbol{x}_i^\top \boldsymbol{\beta} - b_i)^2 \ .$$

$\Longrightarrow n^{O(d)}$-time algorithm with approximation ratio $d+1$,

or $n^{\tilde{O}(d/\epsilon)}$-time algorithm with approximation ratio $1+\epsilon$.

Beating brute-force search: general case

General case: solution may not be determined by only d points.

But, for any RHS $\boldsymbol{b} \in \mathbb{R}^n$, there exist $\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \ldots, \boldsymbol{x}_{i_d}$ s.t. every $\hat{\boldsymbol{\beta}} \in \arg\min_{\boldsymbol{\beta} \in \mathbb{R}^d} \sum_{j=1}^{d} (\boldsymbol{x}_{i_j}^\top \boldsymbol{\beta} - b_{i_j})^2$ satisfies

$$\sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}} - b_i\right)^2 \;\leq\; (d+1) \cdot \min_{\boldsymbol{\beta} \in \mathbb{R}^d} \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{\beta} - b_i\right)^2 .$$

$\Longrightarrow n^{O(d)}$-time algorithm with approximation ratio $d+1$,

or $n^{\tilde{O}(d/\epsilon)}$-time algorithm with approximation ratio $1+\epsilon$.

Better way to get $1+\epsilon$: exploit first-order optimality conditions (i.e., "normal equations") and ϵ-nets.

Overall time: $(n/\epsilon)^{O(k)} + \mathrm{poly}(n,d)$ for $k = \dim(\mathrm{span}(\boldsymbol{x}_i)_{i=1}^n)$.

Lower bound proof sketch

We show that no estimator can confidently distinguish between $\boldsymbol{\beta}^* = \boldsymbol{e}_1$ and $\boldsymbol{\beta}^* = -\boldsymbol{e}_1$, where $\boldsymbol{e}_1 = (1, 0, \ldots, 0)^\top$.

Lower bound proof sketch

We show that no estimator can confidently distinguish between $\boldsymbol{\beta}^* = \boldsymbol{e}_1$ and $\boldsymbol{\beta}^* = -\boldsymbol{e}_1$, where $\boldsymbol{e}_1 = (1, 0, \ldots, 0)^\top$.

Let $P_{\boldsymbol{\beta}^*}$ be the data distribution with parameter $\boldsymbol{\beta}^* \in \{\boldsymbol{e}_1, -\boldsymbol{e}_1\}$.

Task: show $P_{\boldsymbol{e}_1}$ and $P_{-\boldsymbol{e}_1}$ are "close", then appeal to Le Cam's standard "two-point argument":

$$\max_{\boldsymbol{\beta}^* \in \{\boldsymbol{e}_1, -\boldsymbol{e}_1\}} \mathbb{E}_{P_{\boldsymbol{\beta}^*}} \|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2 \ \geq \ 1 - \|P_{\boldsymbol{e}_1} - P_{-\boldsymbol{e}_1}\|_{\mathsf{tv}} \,.$$

Lower bound proof sketch

We show that no estimator can confidently distinguish between $\boldsymbol{\beta}^* = \boldsymbol{e}_1$ and $\boldsymbol{\beta}^* = -\boldsymbol{e}_1$, where $\boldsymbol{e}_1 = (1, 0, \ldots, 0)^\top$.

Let $P_{\boldsymbol{\beta}^*}$ be the data distribution with parameter $\boldsymbol{\beta}^* \in \{\boldsymbol{e}_1, -\boldsymbol{e}_1\}$.

Task: show $P_{\boldsymbol{e}_1}$ and $P_{-\boldsymbol{e}_1}$ are "close", then appeal to Le Cam's standard "two-point argument":

$$\max_{\boldsymbol{\beta}^* \in \{\boldsymbol{e}_1, -\boldsymbol{e}_1\}} \mathbb{E}_{P_{\boldsymbol{\beta}^*}} \|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2 \;\geq\; 1 - \|P_{\boldsymbol{e}_1} - P_{-\boldsymbol{e}_1}\|_{\mathsf{tv}}\,.$$

Key idea: conditional means of $\wr y_i \int_{i=1}^n$ given $(\boldsymbol{x}_i)_{i=1}^n$, under $P_{\boldsymbol{e}_1}$ and $P_{-\boldsymbol{e}_1}$, are close *as unordered multi-sets*.

Proof sketch (continued)

Generative process for $P_{\boldsymbol{\beta}^*}$:

Proof sketch (continued)

Generative process for $P_{\boldsymbol{\beta}^*}$:

1. Draw $(\boldsymbol{x}_i)_{i=1}^n \overset{\text{iid}}{\sim} \mathrm{Uniform}([-1,1]^d)$, $(\varepsilon_i)_{i=1}^n \overset{\text{iid}}{\sim} \mathrm{N}(0,\sigma^2)$.

Proof sketch (continued)

Generative process for $P_{\boldsymbol{\beta}^*}$:

1. Draw $(\boldsymbol{x}_i)_{i=1}^n \overset{\text{iid}}{\sim} \mathrm{Uniform}([-1, 1]^d)$, $(\varepsilon_i)_{i=1}^n \overset{\text{iid}}{\sim} \mathrm{N}(0, \sigma^2)$.
2. Set $u_i := \boldsymbol{x}_i^\top \boldsymbol{\beta}^*$ for $i \in [n]$.

Proof sketch (continued)

Generative process for $P_{\boldsymbol{\beta}^*}$:

1. Draw $(\boldsymbol{x}_i)_{i=1}^n \overset{\text{iid}}{\sim} \mathrm{Uniform}([-1,1]^d)$, $(\varepsilon_i)_{i=1}^n \overset{\text{iid}}{\sim} \mathrm{N}(0,\sigma^2)$.
2. Set $u_i := \boldsymbol{x}_i^\top \boldsymbol{\beta}^*$ for $i \in [n]$.
3. Set $y_i := u_{(i)} + \varepsilon_i$ for $i \in [n]$, where $u_{(1)} \le u_{(2)} \le \cdots \le u_{(n)}$.

Proof sketch (continued)

Generative process for $P_{\boldsymbol{\beta}^*}$:

1. Draw $(\boldsymbol{x}_i)_{i=1}^n \overset{\text{iid}}{\sim} \mathrm{Uniform}([-1,1]^d)$, $(\varepsilon_i)_{i=1}^n \overset{\text{iid}}{\sim} \mathrm{N}(0,\sigma^2)$.
2. Set $u_i := \boldsymbol{x}_i^\top \boldsymbol{\beta}^*$ for $i \in [n]$.
3. Set $y_i := u_{(i)} + \varepsilon_i$ for $i \in [n]$, where $u_{(1)} \leq u_{(2)} \leq \cdots \leq u_{(n)}$.

Conditional distribution of $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$ given $(\boldsymbol{x}_i)_{i=1}^n$:

$$\text{Under } P_{\boldsymbol{e}_1}\text{:} \quad \boldsymbol{y} \mid (\boldsymbol{x}_i)_{i=1}^n \ \sim\ \mathrm{N}(\boldsymbol{u}^{\uparrow}, \sigma^2 \boldsymbol{I}_n)$$

$$\text{Under } P_{-\boldsymbol{e}_1}\text{:} \quad \boldsymbol{y} \mid (\boldsymbol{x}_i)_{i=1}^n \ \sim\ \mathrm{N}(-\boldsymbol{u}^{\downarrow}, \sigma^2 \boldsymbol{I}_n)$$

where $\boldsymbol{u}^{\uparrow} = (u_{(1)}, u_{(2)}, \ldots, u_{(n)})$ and $\boldsymbol{u}^{\downarrow} = (u_{(n)}, u_{(n-1)}, \ldots, u_{(1)})$.

Proof sketch (continued)

Generative process for $P_{\boldsymbol{\beta}^*}$:

1. Draw $(\boldsymbol{x}_i)_{i=1}^n \overset{\text{iid}}{\sim} \mathrm{Uniform}([-1,1]^d)$, $(\varepsilon_i)_{i=1}^n \overset{\text{iid}}{\sim} \mathrm{N}(0,\sigma^2)$.
2. Set $u_i := \boldsymbol{x}_i^\top \boldsymbol{\beta}^*$ for $i \in [n]$.
3. Set $y_i := u_{(i)} + \varepsilon_i$ for $i \in [n]$, where $u_{(1)} \le u_{(2)} \le \cdots \le u_{(n)}$.

Conditional distribution of $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$ given $(\boldsymbol{x}_i)_{i=1}^n$:

$$\text{Under } P_{\boldsymbol{e}_1}\text{:} \quad \boldsymbol{y} \mid (\boldsymbol{x}_i)_{i=1}^n \ \sim\ \mathrm{N}(\boldsymbol{u}^\uparrow, \sigma^2 \boldsymbol{I}_n)$$

$$\text{Under } P_{-\boldsymbol{e}_1}\text{:} \quad \boldsymbol{y} \mid (\boldsymbol{x}_i)_{i=1}^n \ \sim\ \mathrm{N}(-\boldsymbol{u}^\downarrow, \sigma^2 \boldsymbol{I}_n)$$

where $\boldsymbol{u}^\uparrow = (u_{(1)}, u_{(2)}, \ldots, u_{(n)})$ and $\boldsymbol{u}^\downarrow = (u_{(n)}, u_{(n-1)}, \ldots, u_{(1)})$.

Data processing: Lose information by going from $\boldsymbol{y}$ to $\wr y_i \int_{i=1}^n$.

Proof sketch (continued)

By data processing inequality,

$$\begin{aligned}
&\mathrm{KL}\left(P_{\boldsymbol{e}_1}(\cdot \mid (\boldsymbol{x}_i)_{i=1}^n), P_{-\boldsymbol{e}_1}(\cdot \mid (\boldsymbol{x}_i)_{i=1}^n)\right) \\
&\quad \leq \quad \mathrm{KL}\left(\mathrm{N}(\boldsymbol{u}^{\uparrow}, \sigma^2 \boldsymbol{I}_n), \mathrm{N}(-\boldsymbol{u}^{\downarrow}, \sigma^2 \boldsymbol{I}_n)\right)
\end{aligned}$$

Proof sketch (continued)

By data processing inequality,

$$
\begin{aligned}
&\mathrm{KL}\left(P_{\boldsymbol{e}_1}(\cdot \mid (\boldsymbol{x}_i)_{i=1}^n), P_{-\boldsymbol{e}_1}(\cdot \mid (\boldsymbol{x}_i)_{i=1}^n)\right) \\
&\quad \leq \quad \mathrm{KL}\left(\mathrm{N}(\boldsymbol{u}^{\uparrow}, \sigma^2 \boldsymbol{I}_n), \mathrm{N}(-\boldsymbol{u}^{\downarrow}, \sigma^2 \boldsymbol{I}_n)\right) \\
&\quad = \quad \frac{\|\boldsymbol{u}^{\uparrow} - (-\boldsymbol{u}^{\downarrow})\|_2^2}{2\sigma^2}
\end{aligned}
$$

Proof sketch (continued)

By data processing inequality,

$$\begin{aligned} &\mathrm{KL}\left(P_{\boldsymbol{e}_1}(\cdot \mid (\boldsymbol{x}_i)_{i=1}^n), P_{-\boldsymbol{e}_1}(\cdot \mid (\boldsymbol{x}_i)_{i=1}^n)\right) \\ \leq\ & \mathrm{KL}\left(\mathrm{N}(\boldsymbol{u}^{\uparrow}, \sigma^2 \boldsymbol{I}_n), \mathrm{N}(-\boldsymbol{u}^{\downarrow}, \sigma^2 \boldsymbol{I}_n)\right) \\ =\ & \frac{\|\boldsymbol{u}^{\uparrow} - (-\boldsymbol{u}^{\downarrow})\|_2^2}{2\sigma^2} = \frac{\mathsf{SNR}}{2} \cdot \|\boldsymbol{u}^{\uparrow} + \boldsymbol{u}^{\downarrow}\|_2^2 . \end{aligned}$$

Proof sketch (continued)

By data processing inequality,

$$\begin{aligned} &\mathrm{KL}\left(P_{\boldsymbol{e}_1}(\cdot \mid (\boldsymbol{x}_i)_{i=1}^n), P_{-\boldsymbol{e}_1}(\cdot \mid (\boldsymbol{x}_i)_{i=1}^n)\right) \\ &\leq \; \mathrm{KL}\left(\mathrm{N}(\boldsymbol{u}^{\uparrow}, \sigma^2 \boldsymbol{I}_n), \mathrm{N}(-\boldsymbol{u}^{\downarrow}, \sigma^2 \boldsymbol{I}_n)\right) \\ &= \; \frac{\|\boldsymbol{u}^{\uparrow} - (-\boldsymbol{u}^{\downarrow})\|_2^2}{2\sigma^2} \; = \; \frac{\mathsf{SNR}}{2} \cdot \|\boldsymbol{u}^{\uparrow} + \boldsymbol{u}^{\downarrow}\|_2^2 \,. \end{aligned}$$

Some computations show that

$$\mathrm{med}\, \|\boldsymbol{u}^{\uparrow} + \boldsymbol{u}^{\downarrow}\|_2^2 \; \leq \; 4 \,.$$

Proof sketch (continued)

By data processing inequality,

$$\begin{aligned}
&\mathrm{KL}\left(P_{\boldsymbol{e}_1}(\cdot \mid (\boldsymbol{x}_i)_{i=1}^n), P_{-\boldsymbol{e}_1}(\cdot \mid (\boldsymbol{x}_i)_{i=1}^n)\right) \\
&\leq \mathrm{KL}\left(\mathrm{N}(\boldsymbol{u}^{\uparrow}, \sigma^2 \boldsymbol{I}_n), \mathrm{N}(-\boldsymbol{u}^{\downarrow}, \sigma^2 \boldsymbol{I}_n)\right) \\
&= \frac{\|\boldsymbol{u}^{\uparrow} - (-\boldsymbol{u}^{\downarrow})\|_2^2}{2\sigma^2} = \frac{\mathsf{SNR}}{2} \cdot \|\boldsymbol{u}^{\uparrow} + \boldsymbol{u}^{\downarrow}\|_2^2 .
\end{aligned}$$

Some computations show that

$$\mathrm{med}\, \|\boldsymbol{u}^{\uparrow} + \boldsymbol{u}^{\downarrow}\|_2^2 \leq 4 .$$

By conditioning + Pinsker's inequality,

$$\|P_{\boldsymbol{e}_1} - P_{-\boldsymbol{e}_1}\|_{\mathsf{tv}} \leq \frac{1}{2} + \frac{1}{2}\,\mathrm{med}\sqrt{\frac{\mathsf{SNR}}{4} \cdot \|\boldsymbol{u}^{\uparrow} + \boldsymbol{u}^{\downarrow}\|_2^2} \leq \frac{1}{2} + \frac{1}{2}\sqrt{\mathsf{SNR}} .$$

□

Result on exact recovery

Theorem (H., Shi, & Sun, 2017)
Fix any $\boldsymbol{\beta}^* \in \mathbb{R}^d$ *and* $\pi^* \in S_n$, *and assume* $n \geq d$. *Suppose* $(\boldsymbol{x}_i)_{i=0}^n$ *are drawn iid from* $\mathrm{N}(0, \boldsymbol{I}_d)$, *and* $(y_i)_{i=0}^n$ *satisfy*

$$y_0 \;=\; \boldsymbol{x}_0^\top \boldsymbol{\beta}^* \,; \qquad y_i \;=\; \boldsymbol{x}_{\pi^*(i)}^\top \boldsymbol{\beta}^* \,, \quad i = 1, \dots, n \,.$$

There is a $\mathrm{poly}(n, d)$*-time*[‡] *algorithm that, given inputs* $(\boldsymbol{x}_i)_{i=0}^n$ *and* $(y_i)_{i=0}^n$, *returns* π^* *and* $\boldsymbol{\beta}^*$ *with high probability.*

[‡] Assuming problem is appropriately discretized.

Reducing subset sum to shortest vector problem

Lagarias & Odlyzko (1983): random instances of Subset Sum *efficiently solvable* when N source numbers chosen independently and u.a.r. from sufficiently wide interval of $\mathbb{Z}$.

Reducing subset sum to shortest vector problem

Lagarias & Odlyzko (1983): random instances of Subset Sum *efficiently solvable* when N source numbers chosen independently and u.a.r. from sufficiently wide interval of $\mathbb{Z}$.

Main idea: (w.h.p.) every incorrect subset will "miss" the target sum T by noticeable amount.

Reducing subset sum to shortest vector problem

Lagarias & Odlyzko (1983): random instances of Subset Sum *efficiently solvable* when N source numbers chosen independently and u.a.r. from sufficiently wide interval of $\mathbb{Z}$.

Main idea: (w.h.p.) every incorrect subset will "miss" the target sum T by noticeable amount.

Reduction: construct lattice basis in $\mathbb{R}^{N+1}$ such that

- correct subset of basis vectors gives short lattice vector $\boldsymbol{v}_\star$;
- any other lattice vector $\not\propto \boldsymbol{v}_\star$ is more than $2^{N/2}$-times longer.

$$\left[\ \boldsymbol{b}_0 \mid \boldsymbol{b}_1 \mid \cdots \mid \boldsymbol{b}_N\ \right] := \left[\begin{array}{c|ccc} 0 & & \boldsymbol{I}_N & \\ \hline MT & -Mc_1 & \cdots & -Mc_N \end{array}\right]$$

for sufficiently large $M > 0$.

Our random subset sum instance

Catch: Our source numbers $c_{i,j} = y_i \boldsymbol{x}_j^\top \boldsymbol{x}_0$ are **not independent**, and **not uniformly distributed** on some wide interval of $\mathbb{Z}$.

Our random subset sum instance

Catch: Our source numbers $c_{i,j} = y_i \boldsymbol{x}_j^\top \boldsymbol{x}_0$ are **not independent**, and **not uniformly distributed** on some wide interval of $\mathbb{Z}$.

- Instead, have some joint density derived from $\mathrm{N}(0, 1)$.

Our random subset sum instance

Catch: Our source numbers $c_{i,j} = y_i \boldsymbol{x}_j^\top \boldsymbol{x}_0$ are **not independent**, and **not uniformly distributed** on some wide interval of $\mathbb{Z}$.

- Instead, have some joint density derived from $\mathrm{N}(0, 1)$.
- To show that Lagarias & Odlyzko reduction still works, need Gaussian anti-concentration for quadratic and quartic forms.

Our random subset sum instance

Catch: Our source numbers $c_{i,j} = y_i \boldsymbol{x}_j^\top \boldsymbol{x}_0$ are **not independent**, and **not uniformly distributed** on some wide interval of $\mathbb{Z}$.

- Instead, have some joint density derived from $\mathrm{N}(0,1)$.
- To show that Lagarias & Odlyzko reduction still works, need Gaussian anti-concentration for quadratic and quartic forms.

  Key lemma: (w.h.p.) for every $\boldsymbol{Z} \in \mathbb{Z}^{d \times d}$ that is not an integer multiple of permutation matrix corresponding to π^*,

$$\left| y_0 - \sum_{i,j} Z_{i,j} \cdot c_{i,j} \right| \geq \frac{1}{2^{\mathrm{poly}(d)}} \cdot \|\boldsymbol{\beta}^*\|_2 .$$