High Dimensional Linear Regression using Lattice Basis Reduction

Ilias Zadik, joint work with David Gamarnik

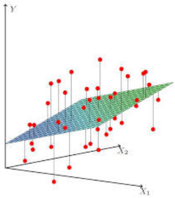
ORC, MIT

Cornell ORIE Young Researchers Workshop, 2018

Linear Regression

Let (unknown) $\beta^* \in \mathbb{R}^p$.
For **measurement matrix** $X \in \mathbb{R}^{n \times p}$, and **noise vector** $W \in \mathbb{R}^n$,
we **observe** n noisy linear samples of β^*, $Y = X\beta^* + W$.

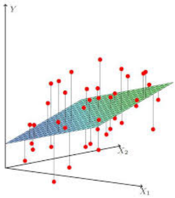
Goal: Given (Y, X), **recover** β^*.



Linear Regression

Let (unknown) $\beta^* \in \mathbb{R}^p$.
For **measurement matrix** $X \in \mathbb{R}^{n \times p}$, and **noise vector** $W \in \mathbb{R}^n$,
we **observe** n noisy linear samples of β^*, $Y = X\beta^* + W$.

Goal: Given (Y, X), **recover** β^*.



Distributional assumption:
X has iid $N(0,1)$ entries and W has iid $N(0,\sigma^2)$ entries.

Main Question:

Question: "What is the **minimum** n (numbers of samples) we need to *efficiently* recover β^*?

Main Question:

Question: "What is the **minimum** n (numbers of samples) we need to *efficiently* recover β^*?

An immediate answer under full generality: at least p.

Main Question:

Question: "What is the **minimum** n (numbers of samples) we need to *efficiently* recover β^*?

An immediate answer under full generality: at least p.

Reason: Even if $\mathrm{W} = 0$, we have $\mathrm{Y} = \mathrm{X}\beta^*$, a linear system with p unknowns and n equations!
To solve it, we need at least p equations, i.e. $\mathrm{n} \geq \mathrm{p}$.

Problem: A High Dimensional Reality

In many *real-life applications* of Linear Regression
(e.g. natural language processing, computer vision, image processing etc)
we observe **much more** features than samples (i.e. $n \ll p$.)

Problem: A High Dimensional Reality

In many *real-life applications* of Linear Regression
(e.g. natural language processing, computer vision, image processing etc)
we observe **much more** features than samples (i.e. $n \ll p$.)

Question: Are we doomed to not use all the features or can we handle such a situation?

Structural Assumptions on β^*

-*Sparsity!* k $\leq$ p non zero coordinates.

- Vast literature.

Structural Assumptions on β^*

-*Sparsity!* $k \leq p$ non zero coordinates.

- Vast literature.
- Requires n of **order** $k \log \left(\frac{pe}{k}\right)$ for known algorithms to work (e.g. Lasso analysis (Wainwright '09))
  Possibly algorithmic hard below (Gamarnik, Z '17), but possible for small noise!

Structural Assumptions on β^*

-*Sparsity!* $k \leq p$ non zero coordinates.

- Vast literature.
- Requires n of **order** $k \log\left(\frac{pe}{k}\right)$ for known algorithms to work (e.g. Lasso analysis (Wainwright '09))
  Possibly algorithmic hard below (Gamarnik, Z '17), but possible for small noise!

-*Ouput of a Generative Model!* (Bora et al '17)
Similar picture: can achieve some $n < p$ but not always small.

Structural Assumptions on β^*

-*Sparsity!* $k \leq p$ non zero coordinates.

- Vast literature.
- Requires n of **order** $k \log \left(\frac{pe}{k}\right)$ for known algorithms to work (e.g. Lasso analysis (Wainwright '09))
  Possibly algorithmic hard below (Gamarnik, Z '17), but possible for small noise!

-*Ouput of a Generative Model!* (Bora et al '17)
Similar picture: can achieve some $n < p$ but not always small.

Success, but is that n sufficiently small for all applications?
Does this structure leads to the best algorithms?

Main Motivation

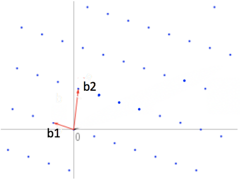
Motivation

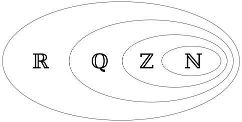
Generic and **natural** assumption that allows

- *efficient* recovery of β^*
- for significantly *small sample sizes.*

This talk

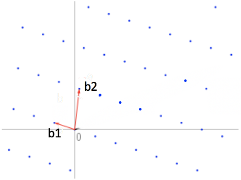
New efficient algorithm for recovering β^* from (Y, X)
based on LLL algorithm (short vectors on lattices)
under a new generic structural assumption (**Q-rationality assumption**).

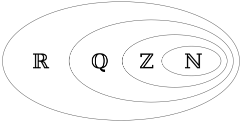




This talk

New efficient algorithm for recovering β^* from (Y, X)
based on LLL algorithm (short vectors on lattices)
under a new generic structural assumption (**Q-rationality assumption**).





Guarantees: works for any n (even n = 1) given sufficiently small noise!

Q-rationality assumption

The Assumption

Every entry of β^* is a **rational number** with denominator Q.

Q-rationality assumption

The Assumption

Every entry of β^* is a **rational number** with denominator Q.

Is it *natural*?

Q-rationality assumption

The Assumption

Every entry of β^* is a **rational number** with denominator Q.

Is it *natural*? Yes!

Q-rationality assumption

The Assumption

Every entry of β^* is a **rational number** with denominator Q.

Is it *natural*? Yes!

(1) Image processing: Each **pixel** has finite many colors (values).

Q-rationality assumption

The Assumption

Every entry of β^* is a **rational number** with denominator Q.

Is it *natural*? Yes!

(1) Image processing: Each **pixel** has finite many colors (values).

(2) GPS [BH '98]: **physics laws** imply for half the coordinates $\beta_i \in \mathbb{Z}$.

Q-rationality assumption

The Assumption

Every entry of β^* is a **rational number** *in* $[0, 1]$ with denominator Q.

Is it *natural*? Yes!

(1) Image processing: Each **pixel** has finite many colors (values).

(2) GPS [BH '98]: **physics laws** imply for half the coordinates $\beta_i \in \mathbb{Z}$.

The New Model

Setup: Let β^* be a **Q-rational** vector. For

- $X \in \mathbb{R}^{n \times p}$ consisting of entries i.i.d $N(0,1)$ **random variables**
- $W \in \mathbb{R}^n$ consisting of entries i.i.d. $N(0,\sigma^2)$ **random variables**

we get n noisy linear samples of β^*, $Y \in \mathbb{R}^n$, given by,

$$Y := X\beta^* + W.$$

The New Model

Setup: Let β^* be a **Q-rational** vector. For

- $\mathsf{X} \in \mathbb{R}^{\mathsf{n} \times \mathsf{p}}$ consisting of entries i.i.d $\mathsf{N}(0,1)$ **random variables**
- $\mathsf{W} \in \mathbb{R}^{\mathsf{n}}$ consisting of entries i.i.d. $\mathsf{N}(0,\sigma^2)$ **random variables**

we get n noisy linear samples of β^*, $\mathsf{Y} \in \mathbb{R}^{\mathsf{n}}$, given by,

$$\mathsf{Y} := \mathsf{X}\beta^* + \mathsf{W}.$$

Goal: Given (Y,X), recover efficiently β^* with n as small as possible. The recovery should happen with probability tending to 1 as p tend to infinity **(w.h.p.)**.

Brute-force with one sample

Any hope for $n = 1$? Recall $y_1 = \langle X_1, \beta^* \rangle + w_1$.

Brute-force with one sample

Any hope for $\mathsf{n} = 1$? Recall $\mathsf{y}_1 = \langle \mathsf{X}_1, \beta^* \rangle + \mathsf{w}_1$.

Yes, if $\sigma = 0$!

Brute-force with one sample

Any hope for $n = 1$? Recall $y_1 = \langle X_1, \beta^* \rangle + w_1$.

Yes, if $\sigma = 0$!

Brute Force Algorithm

Check all Q-rational β for

$$y_1 = \langle X_1, \beta \rangle.$$

Termination Time $\underbrace{(Q+1)(Q+1)\dots(Q+1)}_{p \text{ terms}} = (Q+1)^p$**-not efficient!**

Brute-force with one sample (proof)

Lemma (Brute-force works!)

Suppose β^* Q-*rational and* $\sigma^2 = 0$.
There is no Q-*rational* $\beta \neq \beta^*$ *with* $y_1 = \langle X_1, \beta \rangle$, *almost surely.*

Brute-force with one sample (proof)

Lemma (Brute-force works!)

Suppose β^* Q-*rational and* $\sigma^2 = 0$.
There is no Q-*rational* $\beta \neq \beta^*$ *with* $y_1 = \langle X_1, \beta \rangle$, *almost surely.*

Proof Sketch:

For any $\beta \neq \beta^*$,

$$\mathbb{P}\left(y_1 = \langle X_1, \beta \rangle\right) = \mathbb{P}\left(\langle X_1, \beta^* \rangle = \langle X_1, \beta \rangle\right) = \mathbb{P}\left(\langle X_1, \beta^* - \beta \rangle = 0\right) = 0,$$

since $\langle X_1, \beta^* - \beta \rangle \sim N\left(0, \|\beta^* - \beta\|_2^2\right)$.

Union bound over Q-rational β completes the proof.

An Efficient Algorithm

Theorem (informal, (Gamarnik,Z. NIPS '18))

Suppose you have $\mathsf{n} \ll \mathsf{p}$ *samples and* $0 \leq \sigma \leq \exp\left(-\frac{\mathsf{p}(\mathsf{p}+\log \mathsf{Q})}{2\mathsf{n}}\right)$.
Then there exists a **polynomial-in-**$\mathsf{n}, \mathsf{p}, \log \mathsf{Q}$ *time algorithm which has input* (Y, X) *and ouputs* β^* *w.h.p. as* $\mathsf{p} \to +\infty$.

An Efficient Algorithm

Theorem (informal, (Gamarnik,Z. NIPS '18))

Suppose you have $n \ll p$ *samples and* $0 \leq \sigma \leq \exp\left(-\frac{p(p+\log Q)}{2n}\right)$*.*
Then there exists a **polynomial-in**-$n, p, \log Q$ *time algorithm which has input* (Y, X) *and ouputs* β^* *w.h.p. as* $p \to +\infty$*.*

(1) An efficient algorithm which works for any sample size $n \ll p$.
(2) For $n = 1$, It works in time poly in $p, \log Q$, an exponential decrease from brute-force $(Q+1)^p$.

An Efficient Algorithm

Theorem (informal, (Gamarnik,Z. NIPS '18))

Suppose you have $n \ll p$ *samples and* $0 \leq \sigma \leq \exp\left(-\frac{p(p+\log Q)}{2n}\right)$*.*
Then there exists a **polynomial-in**-$n, p, \log Q$ *time algorithm which has input* (Y, X) *and ouputs* β^* *w.h.p. as* $p \to +\infty$*.*

Call $\sigma_0 = \exp\left(-\frac{p(p+\log Q)}{2n}\right)$. Is this the optimal amount of noise?

An Efficient Algorithm

Theorem (informal, (Gamarnik,Z. NIPS '18))

Suppose you have $n \ll p$ *samples and* $0 \leq \sigma \leq \exp\left(-\frac{p(p+\log Q)}{2n}\right)$.
Then there exists a **polynomial-in**-$n, p, \log Q$ *time algorithm which has input* (Y, X) *and ouputs* β^* *w.h.p. as* $p \to +\infty$.

Call $\sigma_0 = \exp\left(-\frac{p(p+\log Q)}{2n}\right)$. Is this the optimal amount of noise?

Theorem (informal, (Gamarnik,Z. NIPS '18))

Let $Q > 2^p$. *Suppose you have* $n \ll p$ *samples and,* $\sigma > \sigma_0$.
Then its impossible to w.h.p. recover correctly β^* *with* **any algorithm** *with only access to* (Y, X).

An Efficient Algorithm

Theorem (informal, (Gamarnik,Z. NIPS '18))

Suppose you have $\mathrm{n} \ll \mathrm{p}$ *samples and* $0 \leq \sigma \leq \exp\left(-\frac{\mathrm{p}(\mathrm{p}+\log \mathrm{Q})}{2\mathrm{n}}\right)$.
Then there exists a **polynomial-in-**$\mathrm{n}, \mathrm{p}, \log \mathrm{Q}$ *time algorithm which has input* (Y, X) *and ouputs* β^* *w.h.p. as* $\mathrm{p} \to +\infty$.

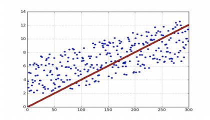
Theorem (informal, (Gamarnik,Z. NIPS '18))

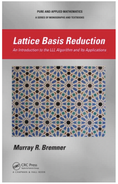
Let $\mathrm{Q} > 2^{\mathrm{p}}$. *Suppose you have* $\mathrm{n} \ll \mathrm{p}$ *samples and,* $\sigma > \sigma_0$.
Then its impossible to w.h.p. recover correctly β^* *with* **any algorithm** *with only access to* (Y, X).

(1) The algorithm has optimal (exponentially small) noise tolerance in the 'high' Q regime!

(2) $\mathrm{Q} = 2^{\mathrm{p}}$ means p bits per coordinate of β^*, reasonable regime.

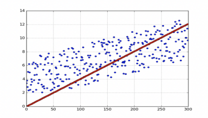
The Algorithm: Connecting HDLR with Lattices

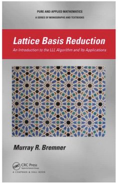




Key Influence:
The algorithm LLL and the use by [Lagarias, Odlyzko '83] and [Frieze '84] for solving subset-sum problems.

The Algorithm: Connecting HDLR with Lattices

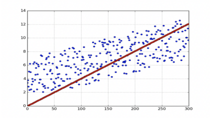


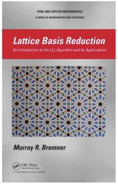


Key Influence:
The algorithm LLL and the use by [Lagarias, Odlyzko '83] and [Frieze '84] for solving subset-sum problems.

Plan

The Algorithm: Connecting HDLR with Lattices



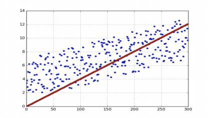


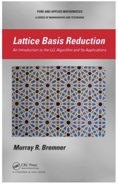
Key Influence:
The algorithm LLL and the use by [Lagarias, Odlyzko '83] and [Frieze '84] for solving subset-sum problems.

Plan

(1) Describe LLL algorithm and general procedure

The Algorithm: Connecting HDLR with Lattices





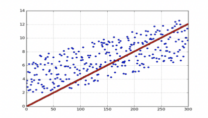
Key Influence:
The algorithm LLL and the use by [Lagarias, Odlyzko '83] and [Frieze '84] for solving subset-sum problems.

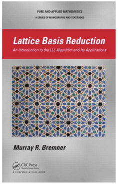
Plan

(1) Describe LLL algorithm and general procedure

(2) Details for using LLL to find β^* in some restrictive case

The Algorithm: Connecting HDLR with Lattices





Key Influence:
The algorithm LLL and the use by [Lagarias, Odlyzko '83] and [Frieze '84] for solving subset-sum problems.

Plan

(1) Describe LLL algorithm and general procedure

(2) Details for using LLL to find β^* in some restrictive case

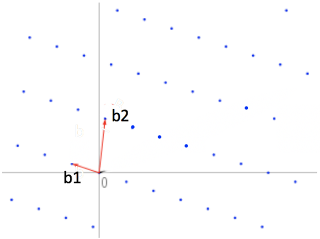
(3) Hint for the general case

Lattices

Let $b_1, \dots, b_m \in \mathbb{Z}^p$ linearly independent vectors.

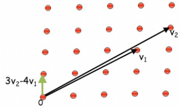
Definition

The lattice $\mathcal{L}$ spanned by $b_1, \dots, b_m$ is the set of all integer combinations of the m vectors.



The LLL algorithm

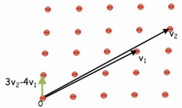
"The Shortest vector problem" $\min_{x \in \mathcal{L} \setminus \{0\}} \|x\|_2$



Shortest Vector Problem (SVP): given a lattice, find a shortest (nonzero) vector

The LLL algorithm

"The Shortest vector problem" $\min_{x \in \mathcal{L} \setminus \{0\}} \|x\|_2$



Shortest Vector Problem (SVP): given a lattice, find a shortest (nonzero) vector

Comments:

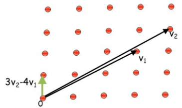
- Generally NP-hard but,
- A famous algorithm proposed by Lenstra-Lenstra-Lovasz (LLL) efficiently approximates it; find $\hat{x} \in \mathcal{L} \setminus \{0\}$ with

$$\|\hat{x}\|_2 \leq 2^{\frac{p}{2}} \min_{x \in \mathcal{L} \setminus \{0\}} \|x\|_2.$$

Time poly in $p, \max\{\|b_i\|_\infty\}$

The LLL algorithm

"The Shortest vector problem" $\min_{x \in \mathcal{L} \setminus \{0\}} \|x\|_2$



Shortest Vector Problem (SVP): given a lattice, find a shortest (nonzero) vector

Comments:

- Generally NP-hard but,
- A famous algorithm proposed by Lenstra-Lenstra-Lovasz (LLL) efficiently approximates it; find $\hat{x} \in \mathcal{L} \setminus \{0\}$ with

$$\|\hat{x}\|_2 \leq 2^{\frac{p}{2}} \min_{x \in \mathcal{L} \setminus \{0\}} \|x\|_2.$$

  Time poly in $p, \max\{\|b_i\|_\infty\}$

  Seems *terrible* but it is **not!!**

Use LLL for Finding β^*

Main Steps:

Use LLL for Finding β^*

Main Steps:

- Create a lattice $\mathcal{L} = \mathcal{L}(\mathsf{Y}, \mathsf{X})$ such that

"approximately" shortest vectors of $\mathcal{L} \leftrightarrow$ "approximately" β^*

Use LLL for Finding β^*

Main Steps:

- Create a lattice $\mathcal{L} = \mathcal{L}(\mathsf{Y}, \mathsf{X})$ such that

"approximately" shortest vectors of $\mathcal{L} \leftrightarrow$ "approximately" β^*

- Use LLL and recover an "approximation" of β^*.

Use LLL for Finding β^*

Main Steps:

- Create a lattice $\mathcal{L} = \mathcal{L}(\mathrm{Y}, \mathrm{X})$ such that

"approximately" shortest vectors of $\mathcal{L} \leftrightarrow$ "approximately" β^*

- Use LLL and recover an "approximation" of β^*.
- Recover β^* from approximation using the structure of β^*.

The Algorithm (special case, [F '84])

Assume

- $n = 1$, $\sigma = 0$, β^* binary: $y = \langle X_1, \beta^* \rangle$.
- $X_1 \in \mathbb{Z}^p$ with iid **uniform in** $[2^N]$ **entries** for large N (say $N = p^2$).

The Algorithm (special case, [F '84])

Assume

- $n = 1$, $\sigma = 0$, β^* binary: $y = \langle X_1, \beta^* \rangle$.
- $X_1 \in \mathbb{Z}^p$ with iid **uniform in** $[2^N]$ **entries** for large N (say $N = p^2$).

(1) For M sufficiently large enough set $\mathcal{L}_M(y_1, X_1)$ produced by the columns of

$$A_M := \begin{bmatrix} MX_1 & -My_1 \\ I_{p\times p} & 0 \end{bmatrix}$$

The Algorithm (special case, [F '84])

Assume

- $n = 1$, $\sigma = 0$, β^* binary: $y = \langle X_1, \beta^* \rangle$.
- $X_1 \in \mathbb{Z}^p$ with iid **uniform in** $[2^N]$ **entries** for large N (say $N = p^2$).

(1) For M sufficiently large enough set $\mathcal{L}_M(y_1, X_1)$ produced by the columns of

$$A_M := \begin{bmatrix} MX_1 & -My_1 \\ I_{p\times p} & 0 \end{bmatrix}$$

Lemma: Each $z \in \mathcal{L}_M$, $\|z\|_2 < M$ is a multiple of $\begin{bmatrix} 0 \\ \beta^* \end{bmatrix}$,w.h.p.

The Algorithm (special case, [F '84])

Assume

- $n = 1$, $\sigma = 0$, β^* binary: $y = \langle X_1, \beta^* \rangle$.
- $X_1 \in \mathbb{Z}^p$ with iid **uniform in** $[2^N]$ **entries** for large N (say $N = p^2$).

(1) For M sufficiently large enough set $\mathcal{L}_M(y_1, X_1)$ produced by the columns of

$$A_M := \begin{bmatrix} MX_1 & -My_1 \\ I_{p \times p} & 0 \end{bmatrix}$$

Lemma: Each $z \in \mathcal{L}_M$, $\|z\|_2 < M$ is a multiple of $\begin{bmatrix} 0 \\ \beta^* \end{bmatrix}$,w.h.p.

Intuition:

$$z = A_M \begin{bmatrix} \beta \\ \lambda \end{bmatrix} = \begin{bmatrix} M\langle X_1, \beta \rangle - M\lambda y_1 \\ \beta \end{bmatrix} = \begin{bmatrix} M\langle X_1, \beta - \lambda\beta^* \rangle \\ \beta \end{bmatrix},$$

$\mathbb{P}\,(\text{Lemma is false}) \leq \mathbb{P}\,(\exists \beta \neq \lambda\beta^* : \|\beta\|_2 < M, \langle X_1, \beta - \lambda\beta^* \rangle = 0) \to 0.$

The Algorithm (special case, [F '84])

(2) Choose M appropriately so that LLL gives a multiple of β^*.

The Algorithm (special case, [F '84])

(2) Choose M appropriately so that LLL gives a multiple of β^*.

- Choose $M = 2^{\frac{p^2}{2}}\sqrt{p} + 1$.
- We know $A_M \begin{bmatrix} \beta^* \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ \beta^* \end{bmatrix} \in \mathcal{L}$.
- LLL gives $\hat{x}$ with norm at most $2^{\frac{p^2}{2}} \| \begin{bmatrix} 0 \\ \beta^* \end{bmatrix} \|_2 \leq 2^{\frac{p^2}{2}}\sqrt{p} < M$.
- Using the lemma we are done!

The Algorithm (special case, [F '84])

(2) Choose M appropriately so that LLL gives a multiple of β^*.

- Choose $M = 2^{\frac{p^2}{2}}\sqrt{p} + 1$.
- We know $A_M \begin{bmatrix} \beta^* \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ \beta^* \end{bmatrix} \in \mathcal{L}$.
- LLL gives $\hat{x}$ with norm at most $2^{\frac{p^2}{2}} \| \begin{bmatrix} 0 \\ \beta^* \end{bmatrix} \|_2 \leq 2^{\frac{p^2}{2}}\sqrt{p} < M$.
- Using the lemma we are done!

(3) Rescale to get β^*.

Comments and (hints for) general Algorithm

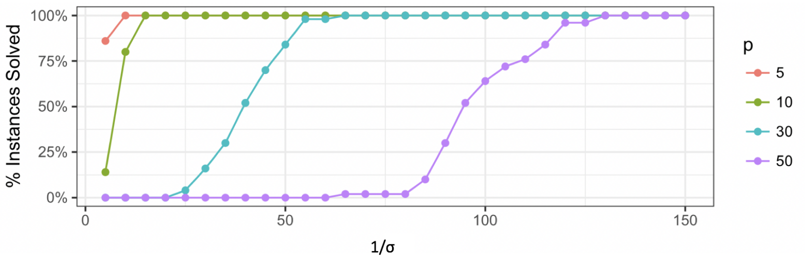
- Reminiscent of the big M-method in LP, but for lattices!

Comments and (hints for) general Algorithm

- Reminiscent of the big M-method in LP, but for lattices!
- Generalizes (after quite some work)
  - From noiseless to noisy measurements.
  - From iid uniform in $[2^N]$ to iid Gaussian ("truncate and multiply")
  - From $n = 1$ to multiple n
  - From binary to Q-rational β.

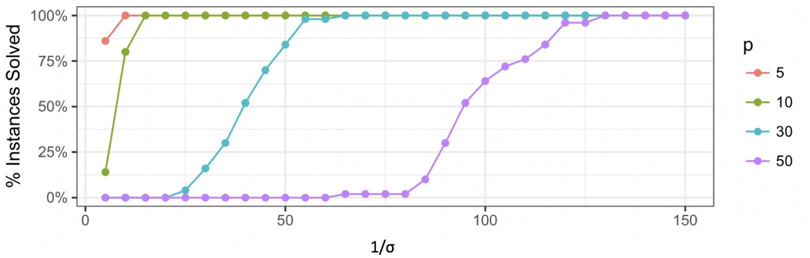
Experimental Results (small p)

(Julia Code by Andrew Zheng and Patricio Foncea (MIT ORC))



Experimental Results (small p)

(Julia Code by Andrew Zheng and Patricio Foncea (MIT ORC))



- Show that algorithm works even for small p!
- Runtime at most 8 minutes (even for $p = 50$)

Summary

(1) We focus on a **new rationality assumption** to perform high dimensional inference for linear regression.

Summary

(1) We focus on a **new rationality assumption** to perform high dimensional inference for linear regression.

(2) Established that β^* is **efficiently recoverable** even when $n = 1$ and noise is small. Also optimal noise tolerance for large Q.

Summary

(1) We focus on a **new rationality assumption** to perform high dimensional inference for linear regression.

(2) Established that β^* is **efficiently recoverable** even when $n = 1$ and noise is small. Also optimal noise tolerance for large Q.

(3) The algorithm is built on an innovative **algorithmic connection** with Lattice theory and Linear Regression.

Summary

(1) We focus on a **new rationality assumption** to perform high dimensional inference for linear regression.

(2) Established that β^* is **efficiently recoverable** even when $n = 1$ and noise is small. Also optimal noise tolerance for large Q.

(3) The algorithm is built on an innovative **algorithmic connection** with Lattice theory and Linear Regression.

(4) **Synthetic experiments** suggest the algorithm works for small p.

Future Directions

(1) Test the algorithms in *real datasets*! Does it perform well?

Future Directions

(1) Test the algorithms in *real datasets*! Does it perform well?

(2) Try the LLL-idea to similar noiseless problems like *Phase Retrieval*! (observe $y_i = |\langle X_i, \beta^* \rangle|$) - ongoing work.

Future Directions

(1) Test the algorithms in *real datasets*! Does it perform well?

(2) Try the LLL-idea to similar noiseless problems like *Phase Retrieval*! (observe $y_i = |\langle X_i, \beta^* \rangle|$) - ongoing work.

(3) Increase the noise level - need something more than just rationality for this.
Idea: use *rationality and sparsity* as an assumption?

Future Directions

(1) Test the algorithms in *real datasets*! Does it perform well?

(2) Try the LLL-idea to similar noiseless problems like *Phase Retrieval*! (observe $y_i = |\langle X_i, \beta^* \rangle|$) - ongoing work.

(3) Increase the noise level - need something more than just rationality for this.
Idea: use *rationality and sparsity* as an assumption?

Thank you!!