Sparse Linear Regression and Lattice Problems

Aparna Gupte

Neekon Vafa

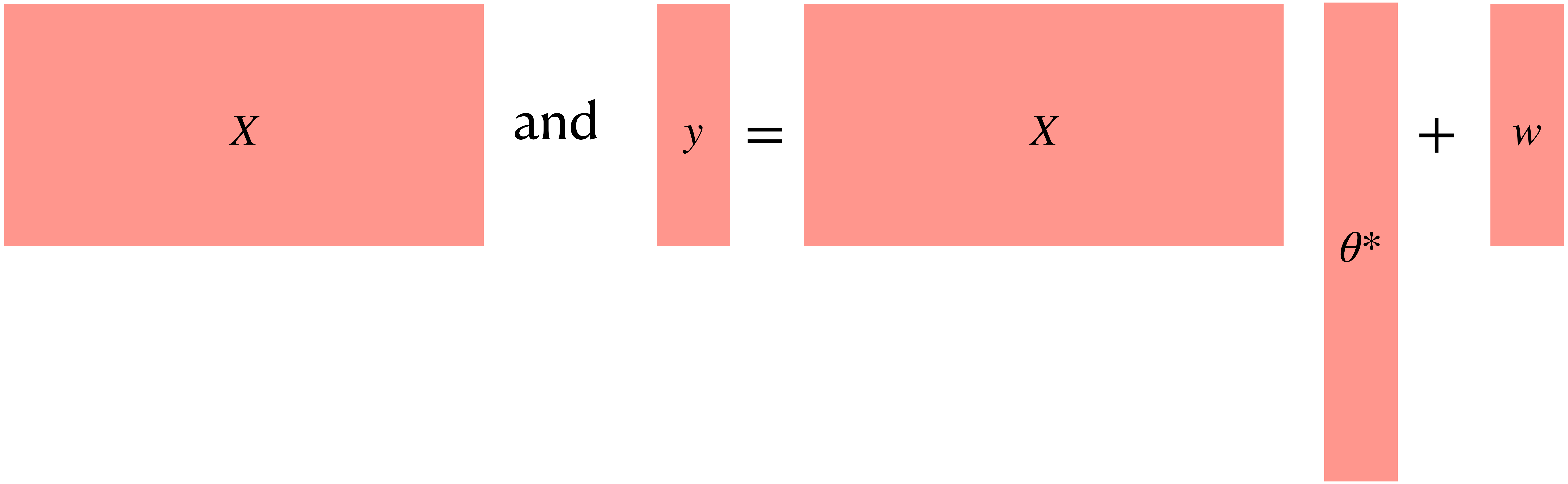
Vinod Vaikuntanathan

Sparse Linear Regression (SLR)

Given: X and y

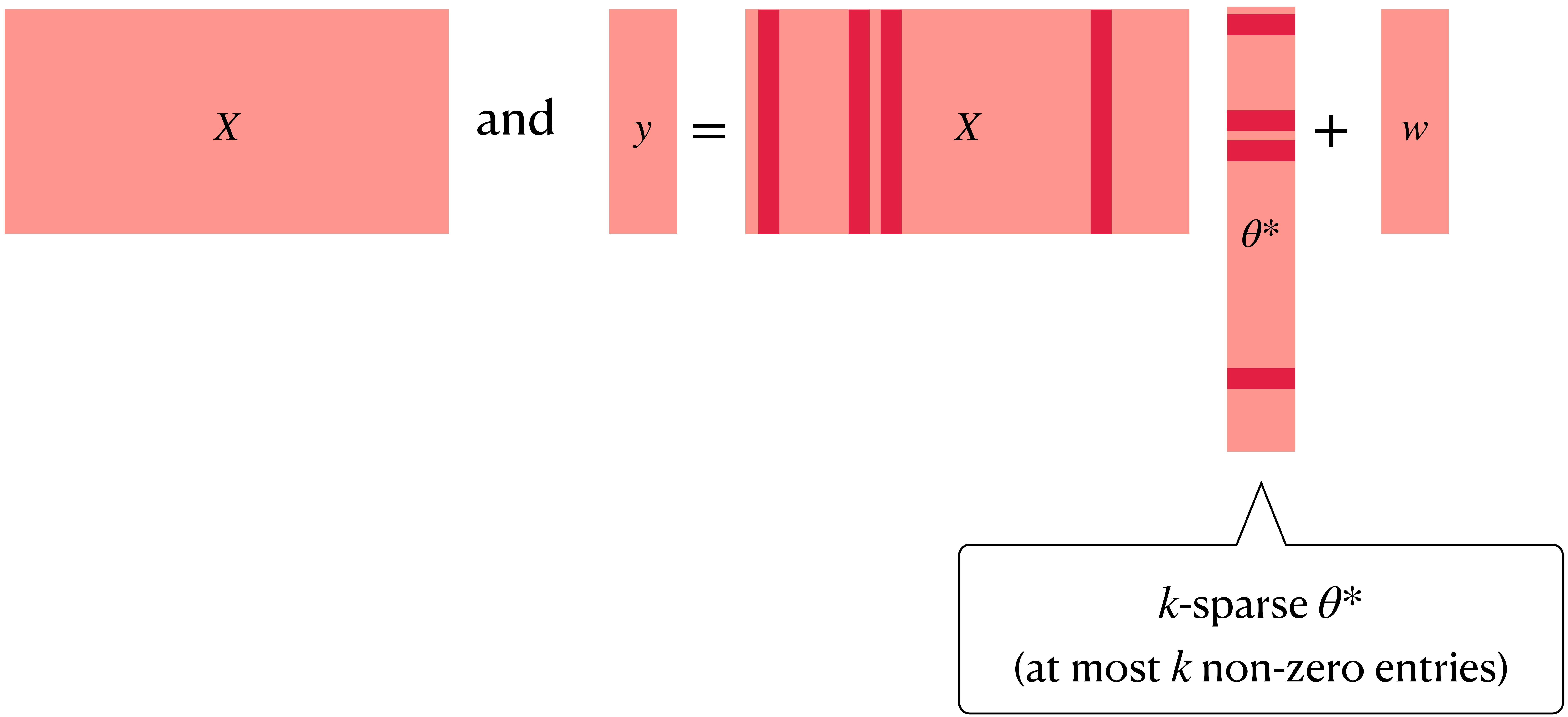
Sparse Linear Regression (SLR)

Given:



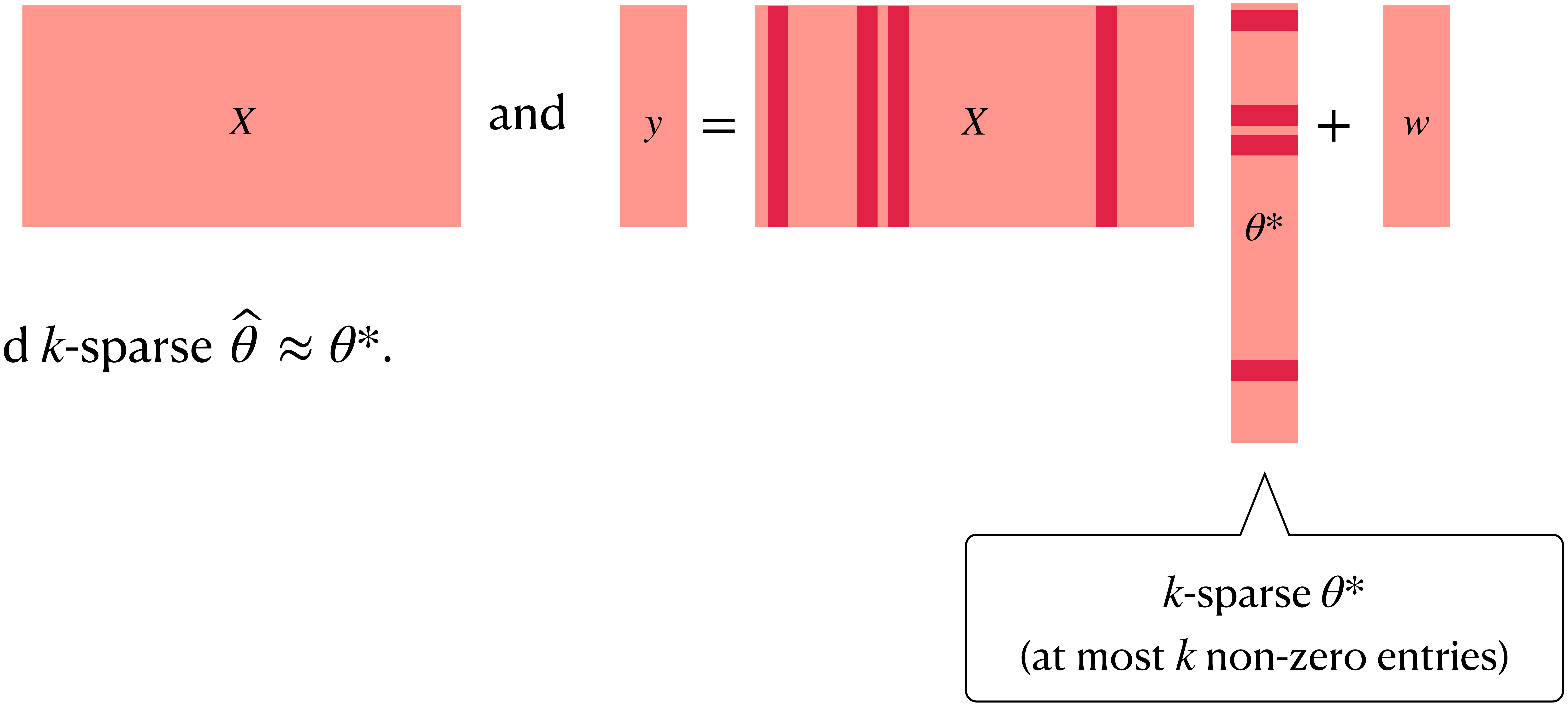
Sparse Linear Regression (SLR)

Given:



Sparse Linear Regression (SLR)

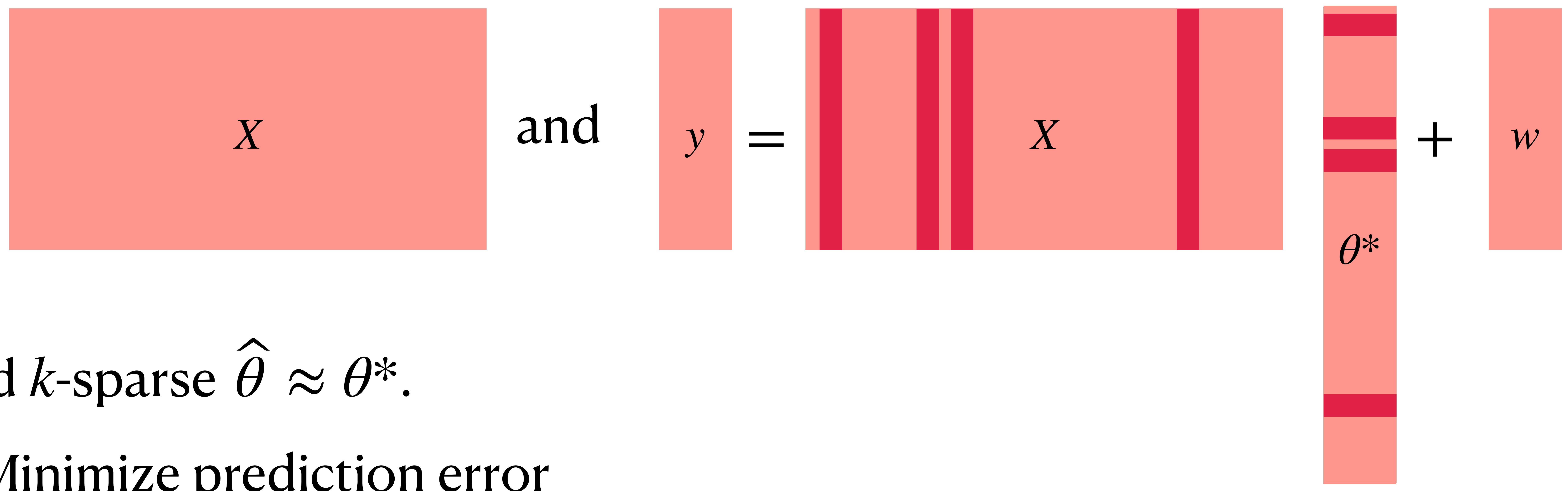
Given:



Goal: Find k-sparse $\widehat{\theta} \approx \theta^*$.

Sparse Linear Regression (SLR)

Given:



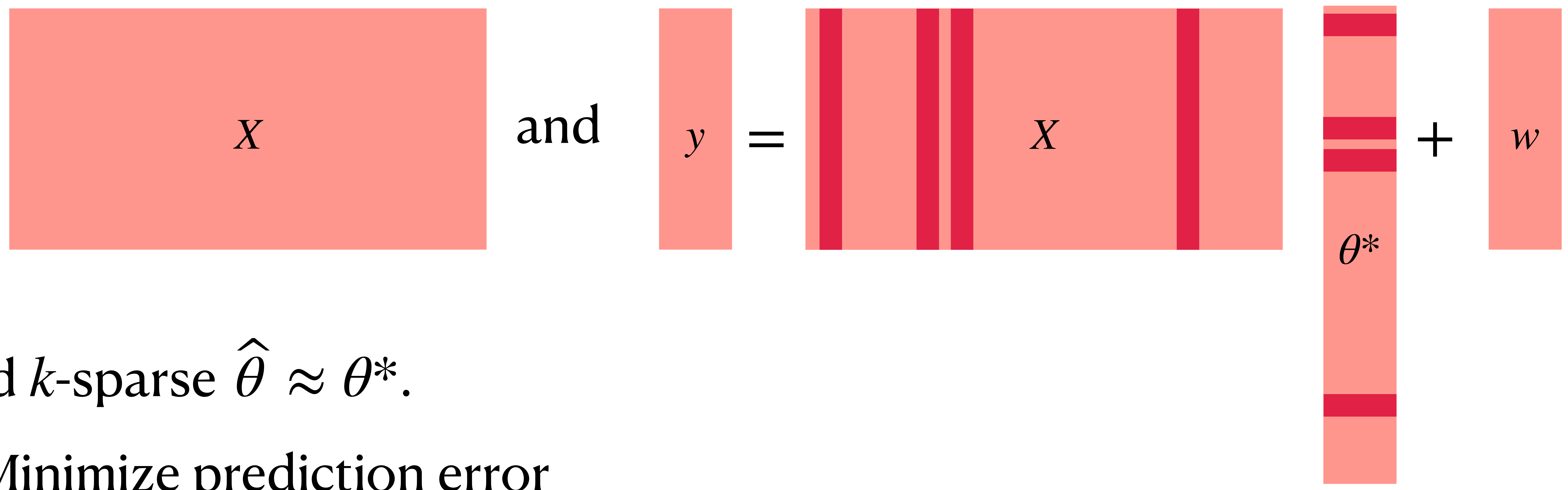
Goal: Find k-sparse $\widehat{\theta} \approx \theta^*$.

Minimize prediction error
$\varepsilon := \|X\widehat{\theta} - X\theta^*\|_2$.

k-sparse θ^*
(at most k non-zero entries)

Sparse Linear Regression (SLR)

Given:



Goal: Find k-sparse $\widehat{\theta} \approx \theta^*$.

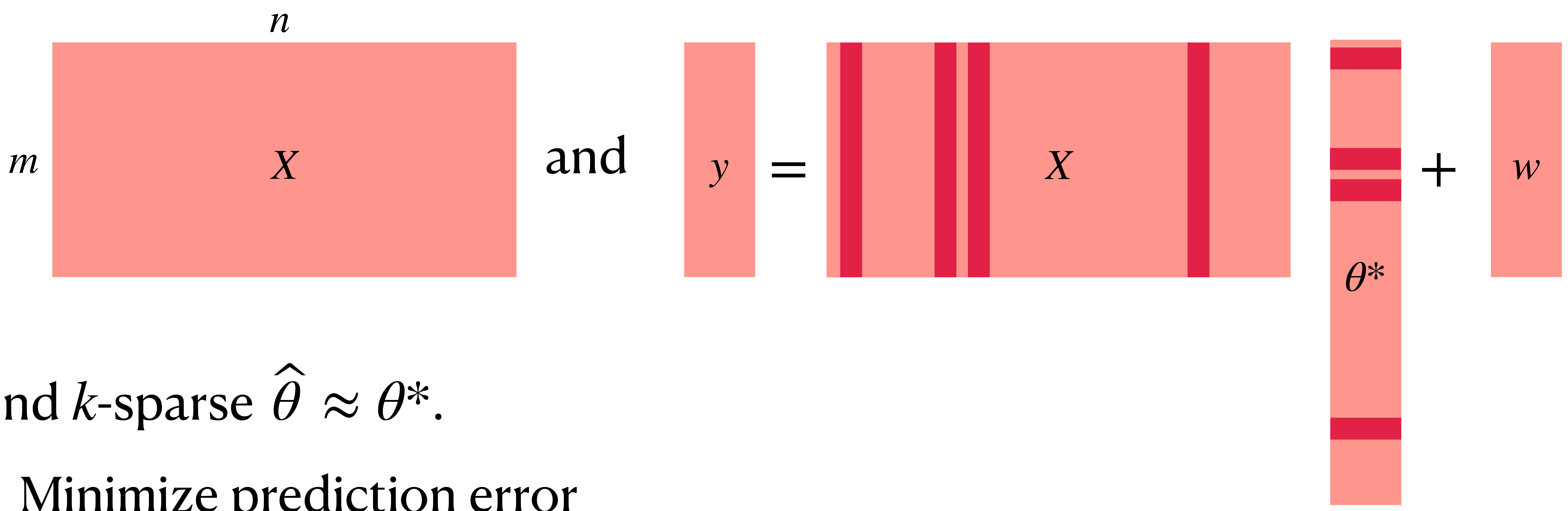
Minimize prediction error
$\varepsilon := \|X\widehat{\theta} - X\theta^*\|_2$.

- Important problem in statistics!

k-sparse θ^*
(at most k non-zero entries)

Sparse Linear Regression (SLR)

Given:



Goal: Find k-sparse $\widehat{\theta} \approx \theta^*$.

Minimize prediction error
$\varepsilon := \|X\widehat{\theta} - X\theta^*\|_2$.

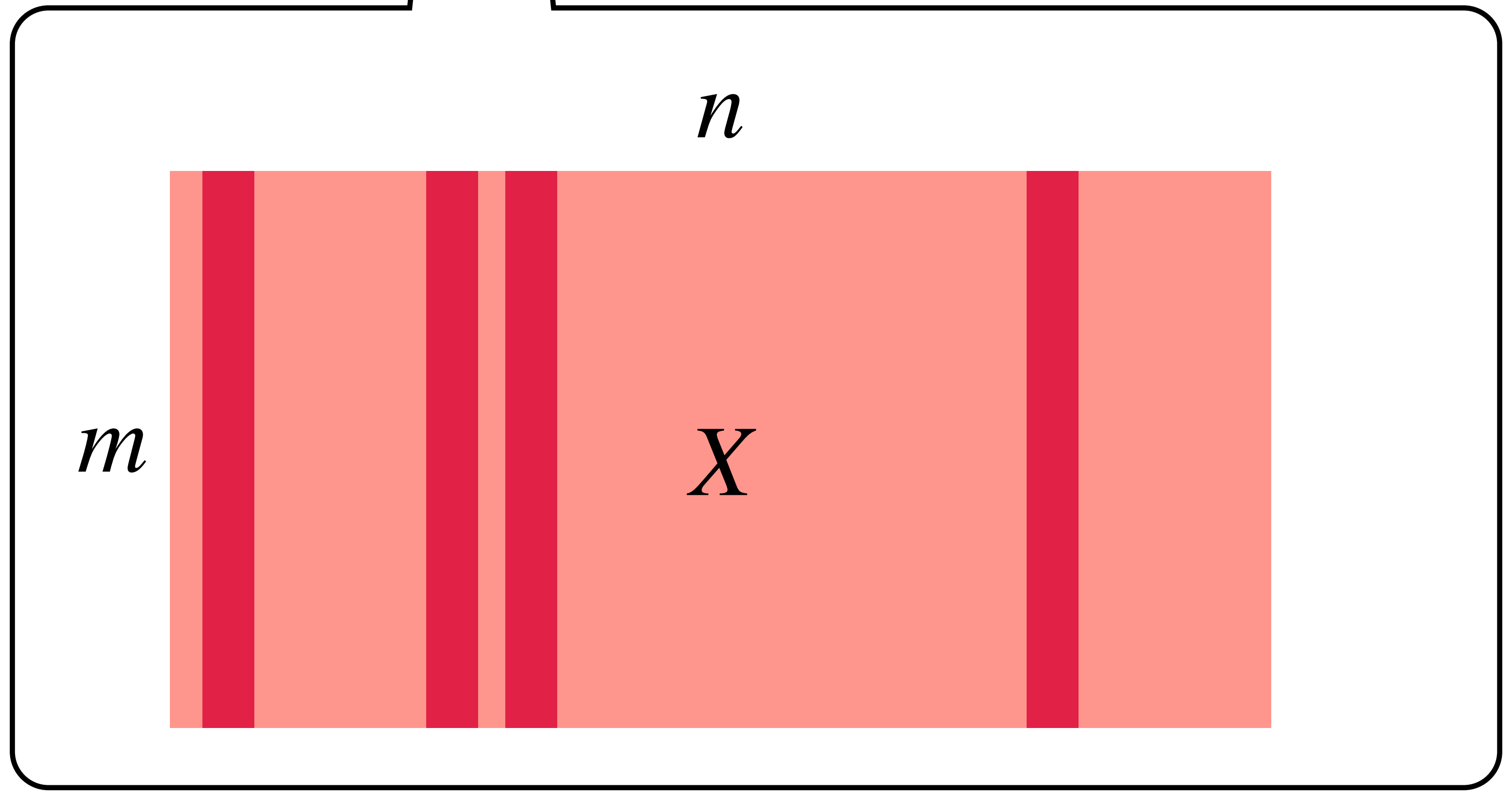
- Important problem in statistics!
- Usually $m \ll n$. Sparsity allows unique θ^*.

k-sparse θ^*
(at most k non-zero entries)

Sparse Linear Regression (SLR)

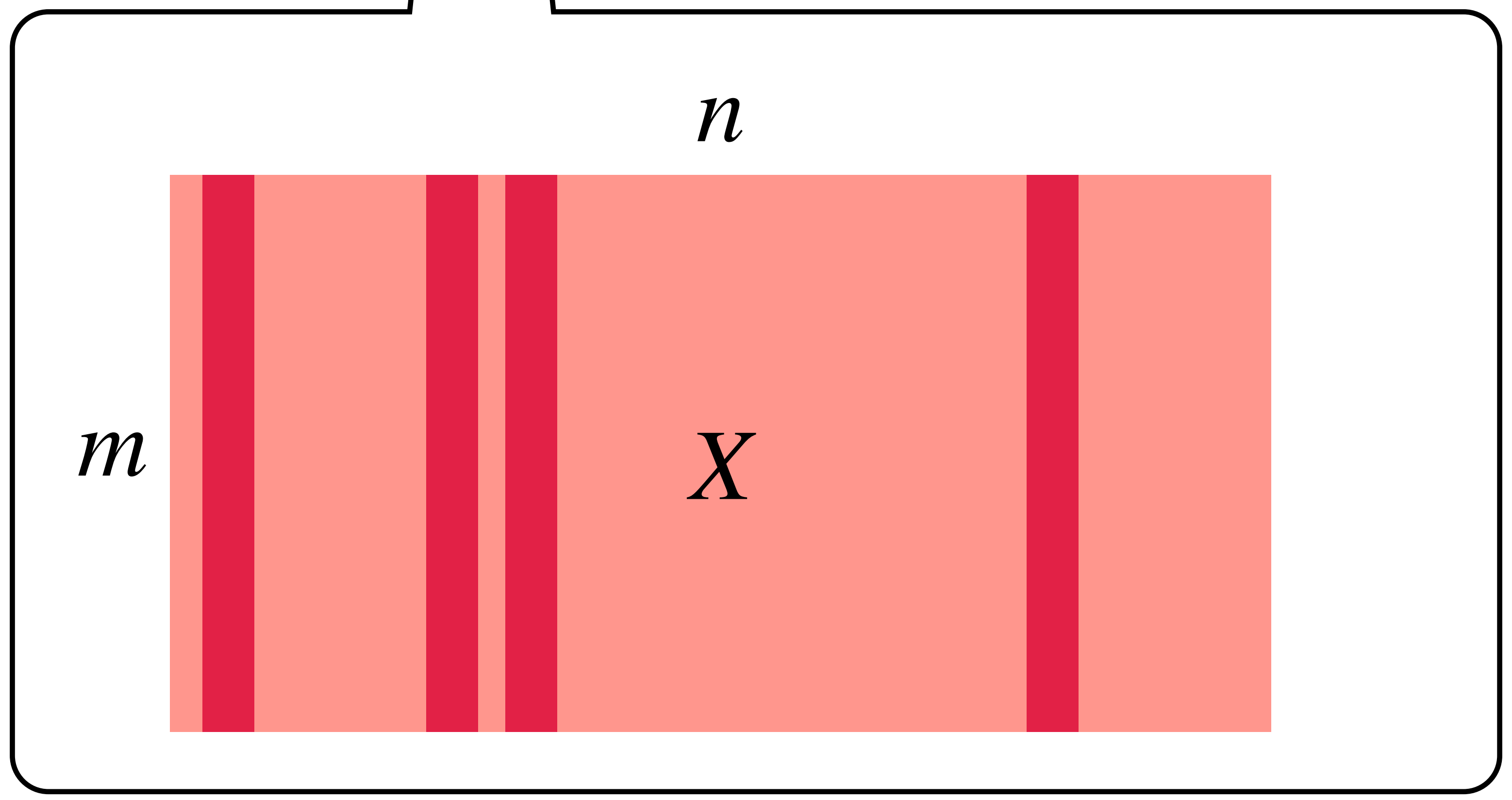
Sparse Linear Regression (SLR)

Algorithms	Description	Runtime	Prediction Error
Naive	Brute force all k-column subsets + least-squares.		



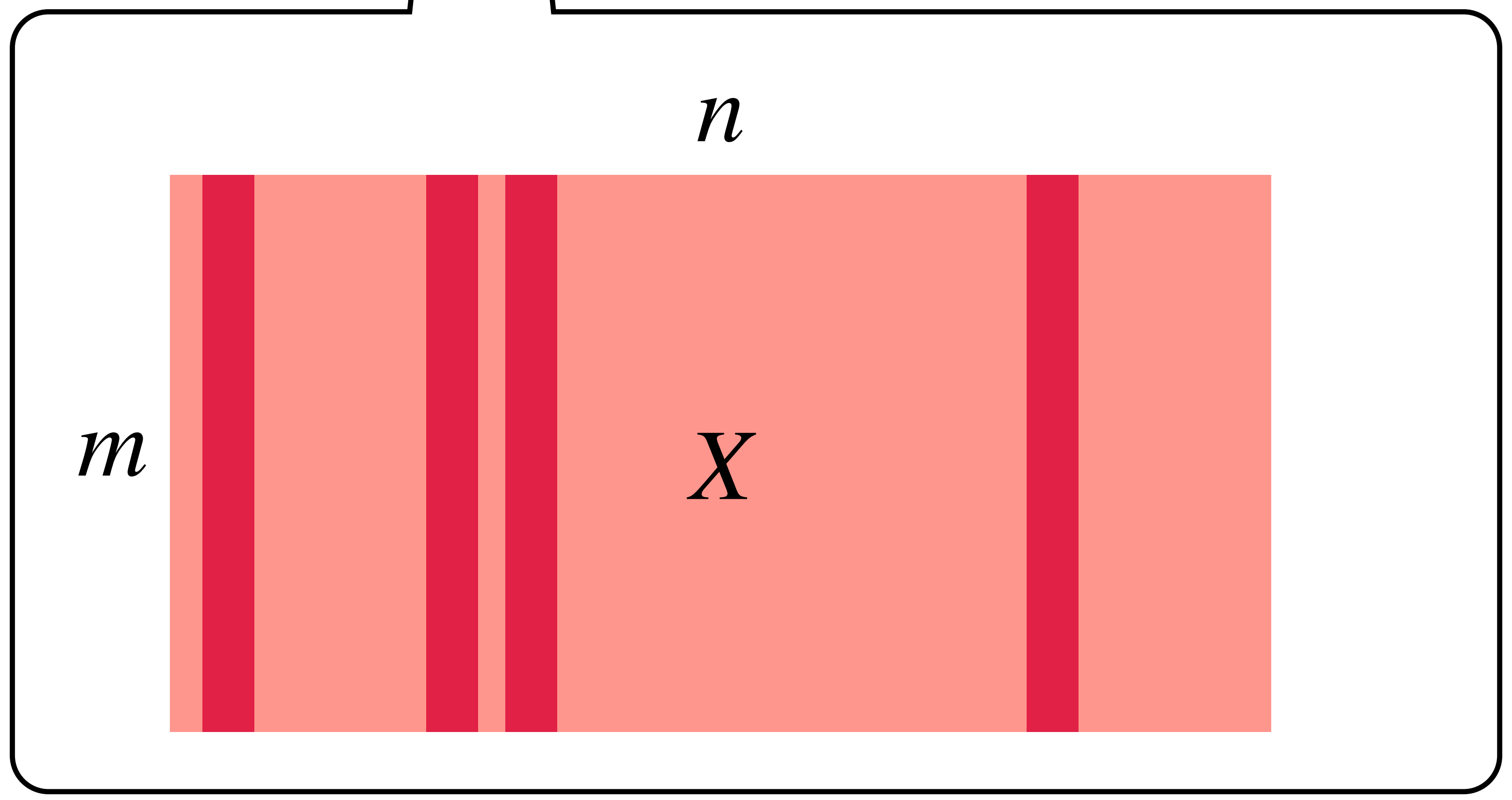
Sparse Linear Regression (SLR)

Algorithms	Description	Runtime	Prediction Error
Naive	Brute force all k-column subsets + least-squares.	n^k 😢	



Sparse Linear Regression (SLR)

Algorithms	Description	Runtime	Prediction Error
Naive	Brute force all k-column subsets + least-squares.	n^k 😢	Optimal prediction error ☺️ $\varepsilon_{\text{opt}} \lesssim 2\|w\|_2$



Sparse Linear Regression (SLR)

Algorithms	Description	Runtime	Prediction Error
Naive	Brute force all k-column subsets + least-squares.	n^k 😢	Optimal prediction error ☺️ $\varepsilon_{\text{opt}} \leq 2\|w\|_2$
LASSO	Want: $\min_{\theta: \|\theta\|_0 \leq k} \|X\theta - y\|_2$		

Sparse Linear Regression (SLR)

Algorithms	Description	Runtime	Prediction Error
Naive	Brute force all k-column subsets + least-squares.	n^k 😢	Optimal prediction error ☺️ $\varepsilon_{\text{opt}} \le 2\|w\|_2$
LASSO	Want: $\min_{\theta:\|\theta\|_0 \le k} \|X\theta - y\|_2$ Equivalently: $\min_{\theta} \|X\theta - y\|_2^2 + \gamma \cdot \|\theta\|_0$		

Sparse Linear Regression (SLR)

Algorithms	Description	Runtime	Prediction Error
Naive	Brute force all k-column subsets + least-squares.	n^k 😢	Optimal prediction error ☺️ $\varepsilon_{\text{opt}} \lesssim 2\|w\|_2$
LASSO	Want: $\min_{\theta:\|\theta\|_0 \leq k} \|X\theta - y\|_2$ Equivalently: $\min_{\theta} \|X\theta - y\|_2^2 + \gamma \cdot \|\theta\|_0$		

Not convex 😢

Sparse Linear Regression (SLR)

Algorithms	Description	Runtime	Prediction Error
Naive	Brute force all k-column subsets + least-squares.	n^k 😢	Optimal prediction error ☺️ $\varepsilon_{\text{opt}} \le 2\lVert w \rVert_2$
LASSO	Want: $\min_{\theta: \lVert\theta\rVert_0 \le k} \lVert X\theta - y \rVert_2$ Equivalently: $\min_{\theta} \lVert X\theta - y \rVert_2^2 + \gamma \cdot$ ~~$\lVert\theta\rVert_0$~~ $\lVert\theta\rVert_1$		

Convex ☺️

Sparse Linear Regression (SLR)

Algorithms	Description	Runtime	Prediction Error
Naive	Brute force all k-column subsets + least-squares.	n^k 😢	Optimal prediction error ☺️ $\varepsilon_{\text{opt}} \lesssim 2\|w\|_2$
LASSO	Want: $\min_{\theta:\|\theta\|_0 \leq k} \|X\theta - y\|_2$ Equivalently: $\min_{\theta} \|X\theta - y\|_2^2 + \gamma \cdot \|\theta\|_{\cancel{0}\,1}$	poly-time ☺️	

Convex ☺️

Sparse Linear Regression (SLR)

Algorithms	Description	Runtime	Prediction Error
Naive	Brute force all k-column subsets + least-squares.	n^k 😢	Optimal prediction error ☺️ $\varepsilon_{\text{opt}} \leq 2\|w\|_2$
LASSO	Want: $\min_{\theta : \|\theta\|_0 \leq k} \|X\theta - y\|_2$ Equivalently: $\min_{\theta} \|X\theta - y\|_2^2 + \gamma \cdot \|\theta\|_{\cancel{0}1}$	poly-time ☺️	Error depends on conditioning of X 😢

Sparse Linear Regression (SLR)

Algorithms	Description	Runtime	Prediction Error
Naive	Brute force all k-column subsets + least-squares.	n^k 😢	Optimal prediction error ☺️ $\varepsilon_{\text{opt}} \leq 2\|w\|_2$
LASSO	Want: $\min_{\theta : \|\theta\|_0 \leq k} \|X\theta - y\|_2$ Equivalently: $\min_{\theta} \|X\theta - y\|_2^2 + \gamma \cdot \|\theta\|_{\cancel{0}1}$	poly-time ☺️	Error depends on conditioning of X 😢 $\varepsilon_{\text{LASSO}} \leq O\left(\|w\|_2 \cdot \frac{k}{\text{RE}(X)}\right)$

Sparse Linear Regression (SLR)

Algorithms	Description	Runtime	Prediction Error
Naive	Brute force all k-column subsets + least-squares.	n^k 😢	Optimal prediction error ☺️ $\varepsilon_{\text{opt}} \lesssim 2\|w\|_2$
LASSO	Want: $\min_{\theta : \|\theta\|_0 \le k} \|X\theta - y\|_2$ Equivalently: $\min_{\theta} \|X\theta - y\|_2^2 + \gamma \cdot \|\theta\|_{\cancel{0}1}$	poly-time ☺️	Error depends on conditioning of X 😢 $\varepsilon_{\text{LASSO}} \le O\left(\|w\|_2 \cdot \frac{k}{\text{RE}(X)}\right)$

Restricted eigenvalue constant *

$$\text{RE}(X) \approx \min_{\text{sparse } \theta} \frac{\|X\theta\|_2}{\|\theta\|_2}$$

* for "normalized" X

Sparse Linear Regression (SLR)

Algorithms	Description	Runtime	Prediction Error
Naive	Brute force all k-column subsets + least-squares.	n^k 😢	Optimal prediction error ☺️ $\varepsilon_{\text{opt}} \lesssim 2\|w\|_2$
LASSO	Want: $\min_{\theta: \|\theta\|_0 \le k} \|X\theta - y\|_2$ Equivalently: $\min_\theta \|X\theta - y\|_2^2 + \gamma \cdot$ ~~$\|\theta\|_0$~~ $\|\theta\|_1$	poly-time ☺️	Error depends on conditioning of X 😢 $\varepsilon_{\text{LASSO}} \le O\left(\|w\|_2 \cdot \frac{k}{\text{RE}(X)}\right)$

When $X \sim \mathcal{N}(0, I_n)^{\otimes m}$ and $m \ge \Omega(k \cdot \log n)$

(i.e. rows i.i.d. standard Gaussian & unique sparse solution),

$\text{RE}(X) = \Omega(1)$, so LASSO achieves $\tilde{O}(\varepsilon_{\text{opt}})$ error*.

Restricted eigenvalue constant *

$$\text{RE}(X) \approx \min_{\text{sparse } \theta} \frac{\|X\theta\|_2}{\|\theta\|_2}$$

* for "normalized" X

* $\tilde{O}$ hides a factor of k

SLR: computational-statistical gap?

Is there a better algorithm for SLR *without the RE(X) dependence?*

SLR: computational-statistical gap?

Is there a better algorithm for SLR *without the RE(X) dependence?*

Hardness [Natarajan '95, Zhang-Wainwright-Jordan '14]

- A fixed sequence of *worst-case* design matrices for which it is impossible to beat the RE error bound unless NP $\subseteq$ P/poly.

SLR: computational-statistical gap?

Is there a better algorithm for SLR *without the RE(X) dependence?*

Hardness [Natarajan '95, Zhang-Wainwright-Jordan '14]

- A fixed sequence of *worst-case* design matrices for which it is impossible to beat the RE error bound unless NP $\subseteq$ P/poly.

Algorithms [Kelner-Koehler-Meka-Rohatgi '22, '23]

- Preconditioned LASSO beats RE bound for certain X (X with "low treewidth").

SLR: computational-statistical gap?

Is there a better algorithm for SLR *without the RE(X) dependence?*

Hardness [Natarajan '95, Zhang-Wainwright-Jordan '14]

- A fixed sequence of *worst-case* design matrices for which it is impossible to beat the RE error bound unless NP $\subseteq$ P/poly.

Algorithms [Kelner-Koehler-Meka-Rohatgi '22, '23]

- Preconditioned LASSO beats RE bound for certain X (X with "low treewidth").

For which design matrices X is SLR hard?

SLR: computational-statistical gap?

Is there a better algorithm for SLR *without the RE*(X) *dependence?*

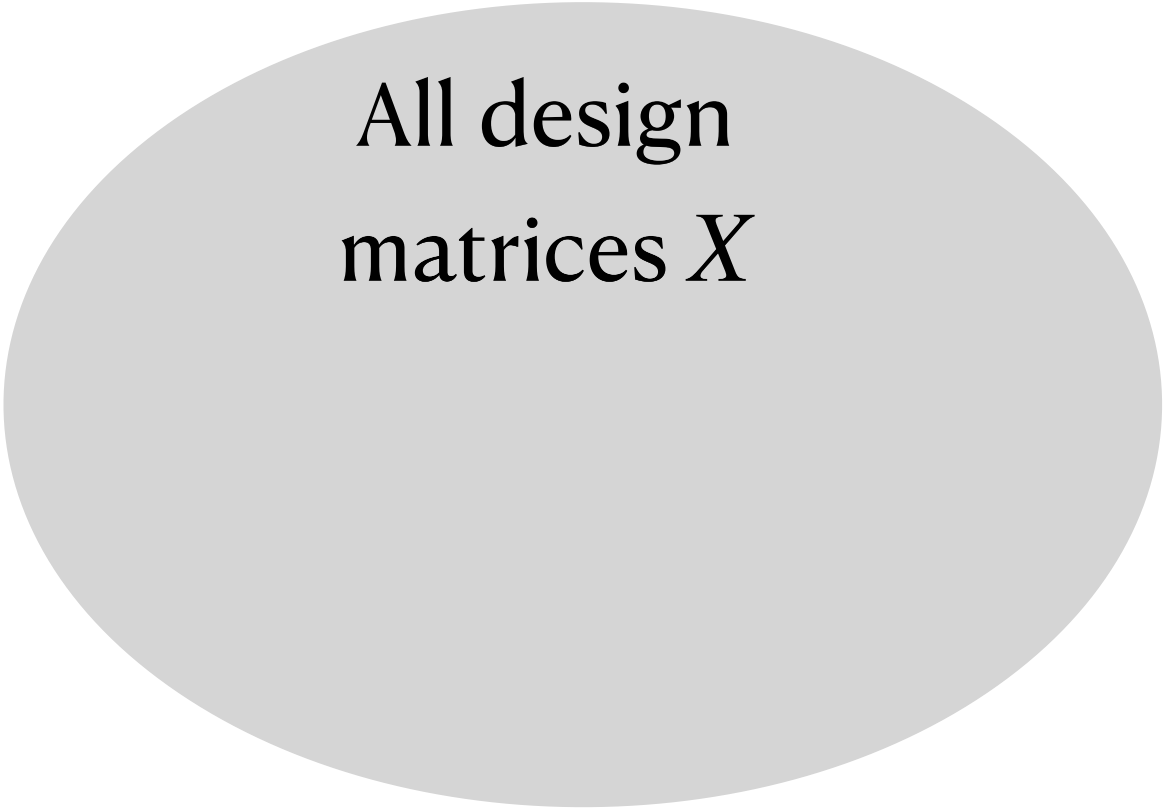
Hardness [Natarajan '95, Zhang-Wainwright-Jordan '14]

- A fixed sequence of *worst-case* design matrices for which it is impossible to beat the RE error bound unless $\text{NP} \subseteq \text{P/poly}$.

Algorithms [Kelner-Koehler-Meka-Rohatgi '22, '23]

- Preconditioned LASSO beats RE bound for certain X (X with "low treewidth").

For which design matrices X is SLR hard?



SLR: computational-statistical gap?

Is there a better algorithm for SLR *without the RE*(X) *dependence?*

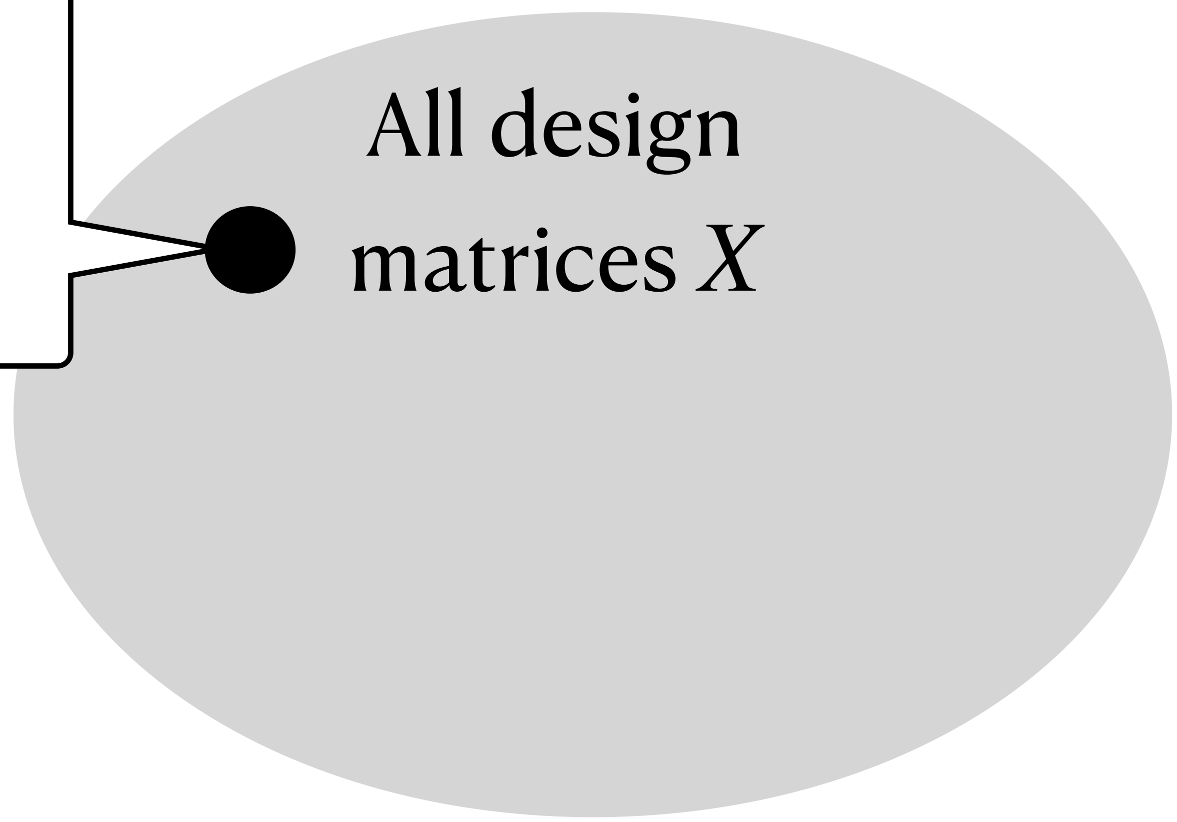
Hardness [Natarajan '95, Zhang-Wainwright-Jordan '14]

- A fixed sequence of *worst-case* design matrices for which it is impossible to beat the RE error bound unless $\mathrm{NP} \subseteq \mathrm{P/poly}$.

Algorithms [Kelner-Koehler-Meka-Rohatgi '22, '23]

- Preconditioned LASSO beats RE bound for certain X (X with "low treewidth").

ZWJ design matrix:
Hard to beat RE bound



For which design matrices X is SLR hard?

SLR: computational-statistical gap?

Is there a better algorithm for SLR *without the RE(X) dependence?*

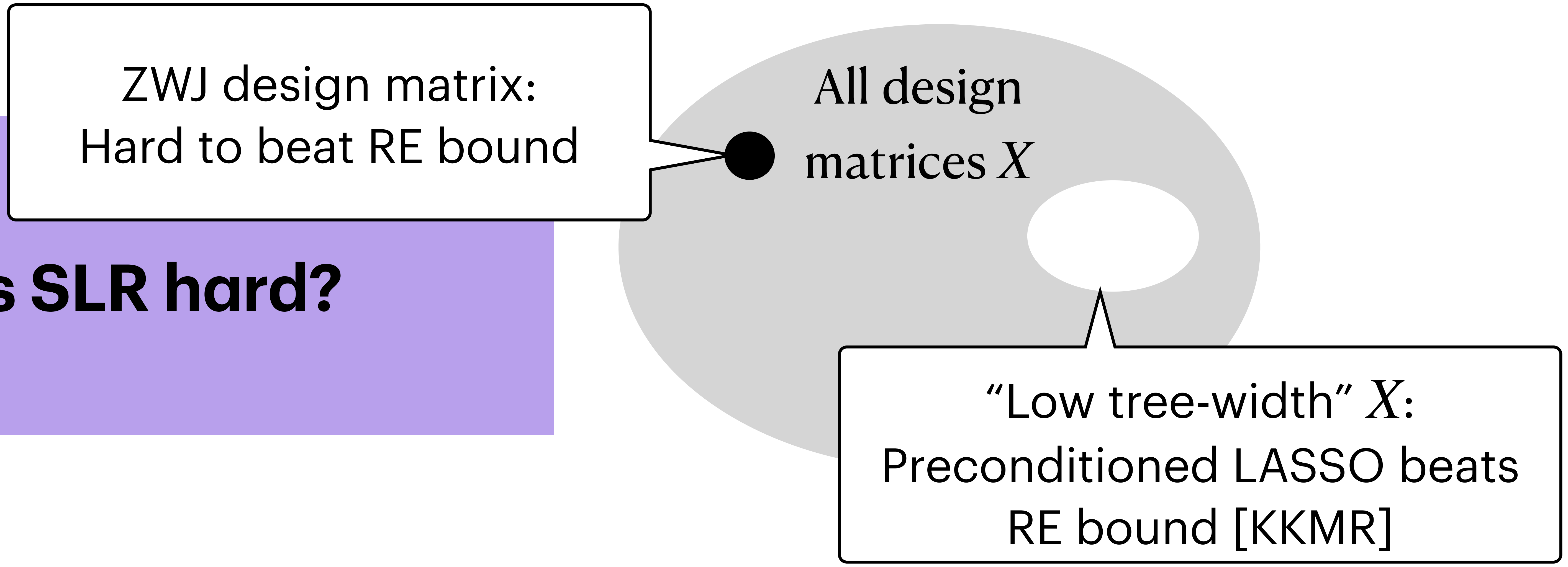
Hardness [Natarajan '95, Zhang-Wainwright-Jordan '14]

- A fixed sequence of *worst-case* design matrices for which it is impossible to beat the RE error bound unless NP $\subseteq$ P/poly.

Algorithms [Kelner-Koehler-Meka-Rohatgi '22, '23]

- Preconditioned LASSO beats RE bound for certain X (X with "low treewidth").

For which design matrices X is SLR hard?



SLR: computational-statistical gap?

Is there a better algorithm for SLR *without the RE(X) dependence?*

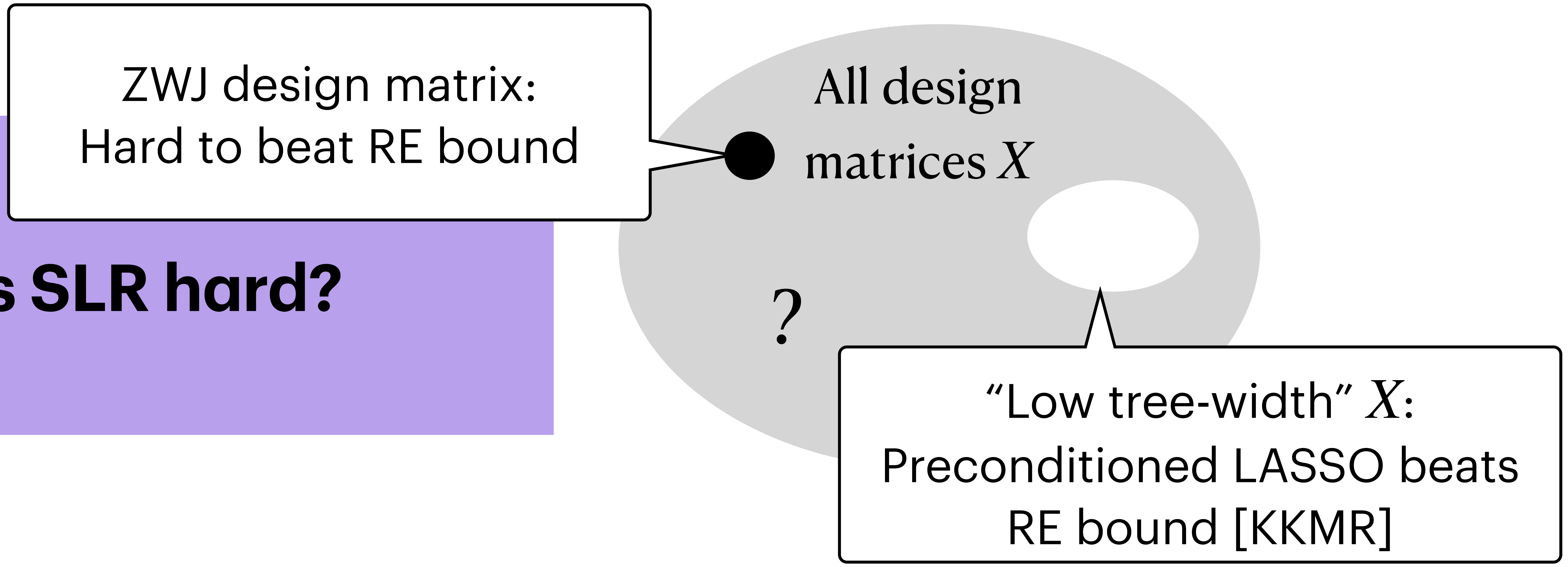
Hardness [Natarajan '95, Zhang-Wainwright-Jordan '14]

- A fixed sequence of *worst-case* design matrices for which it is impossible to beat the RE error bound unless NP $\subseteq$ P/poly.

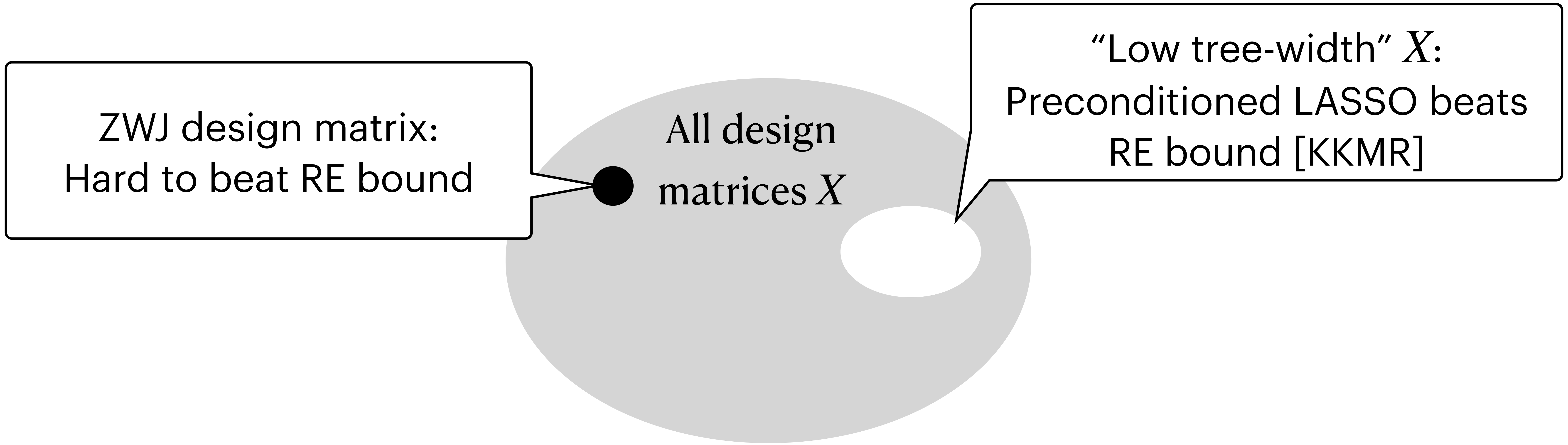
Algorithms [Kelner-Koehler-Meka-Rohatgi '22, '23]

- Preconditioned LASSO beats RE bound for certain X (X with "low treewidth").

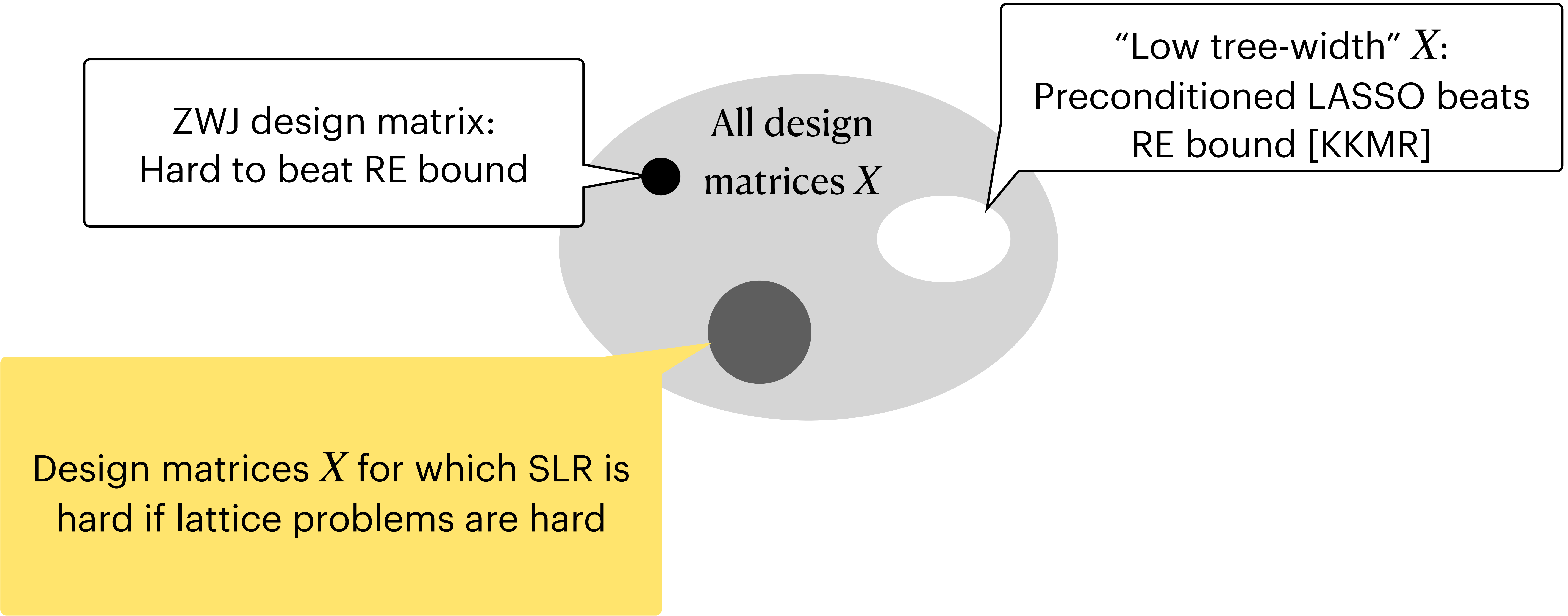
For which design matrices X is SLR hard?



Tl;dr

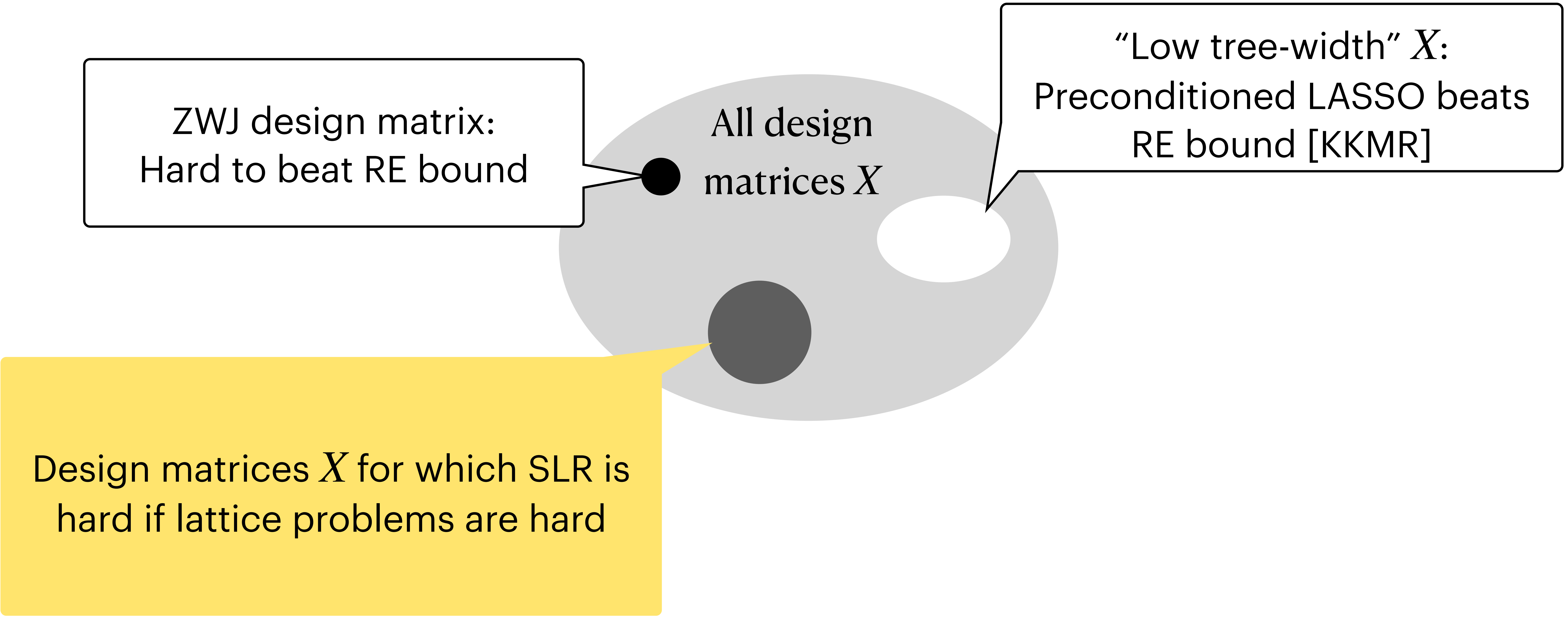

ZWJ design matrix:
Hard to beat RE bound
All design
matrices X
"Low tree-width" X:
Preconditioned LASSO beats
RE bound [KKMR]

Tl;dr

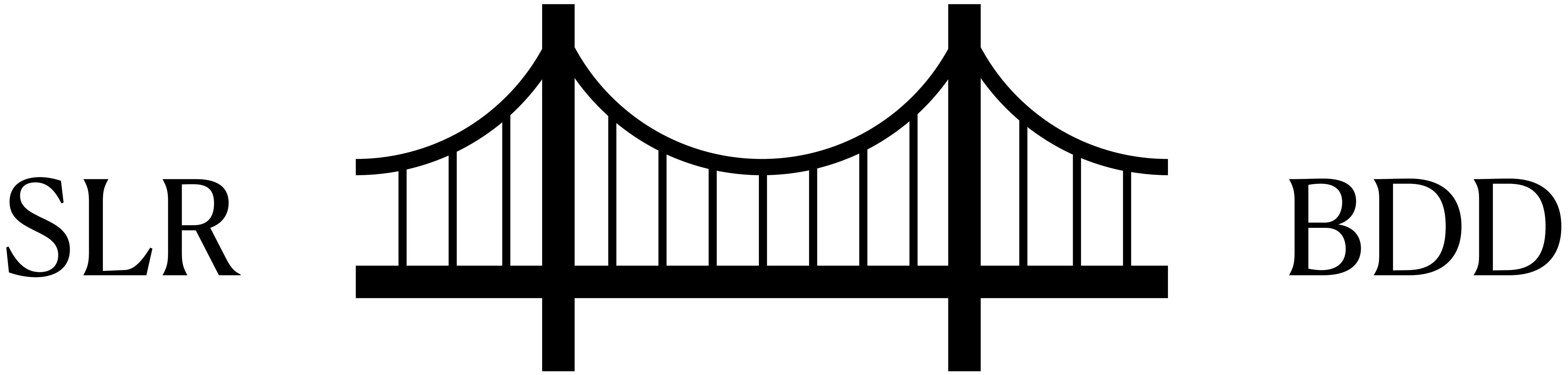


ZWJ design matrix:
Hard to beat RE bound
All design matrices X
"Low tree-width" X:
Preconditioned LASSO beats RE bound [KKMR]
Design matrices X for which SLR is hard if lattice problems are hard

Tl;dr



ZWJ design matrix:
Hard to beat RE bound
All design matrices X
"Low tree-width" X:
Preconditioned LASSO beats RE bound [KKMR]
Design matrices X for which SLR is hard if lattice problems are hard



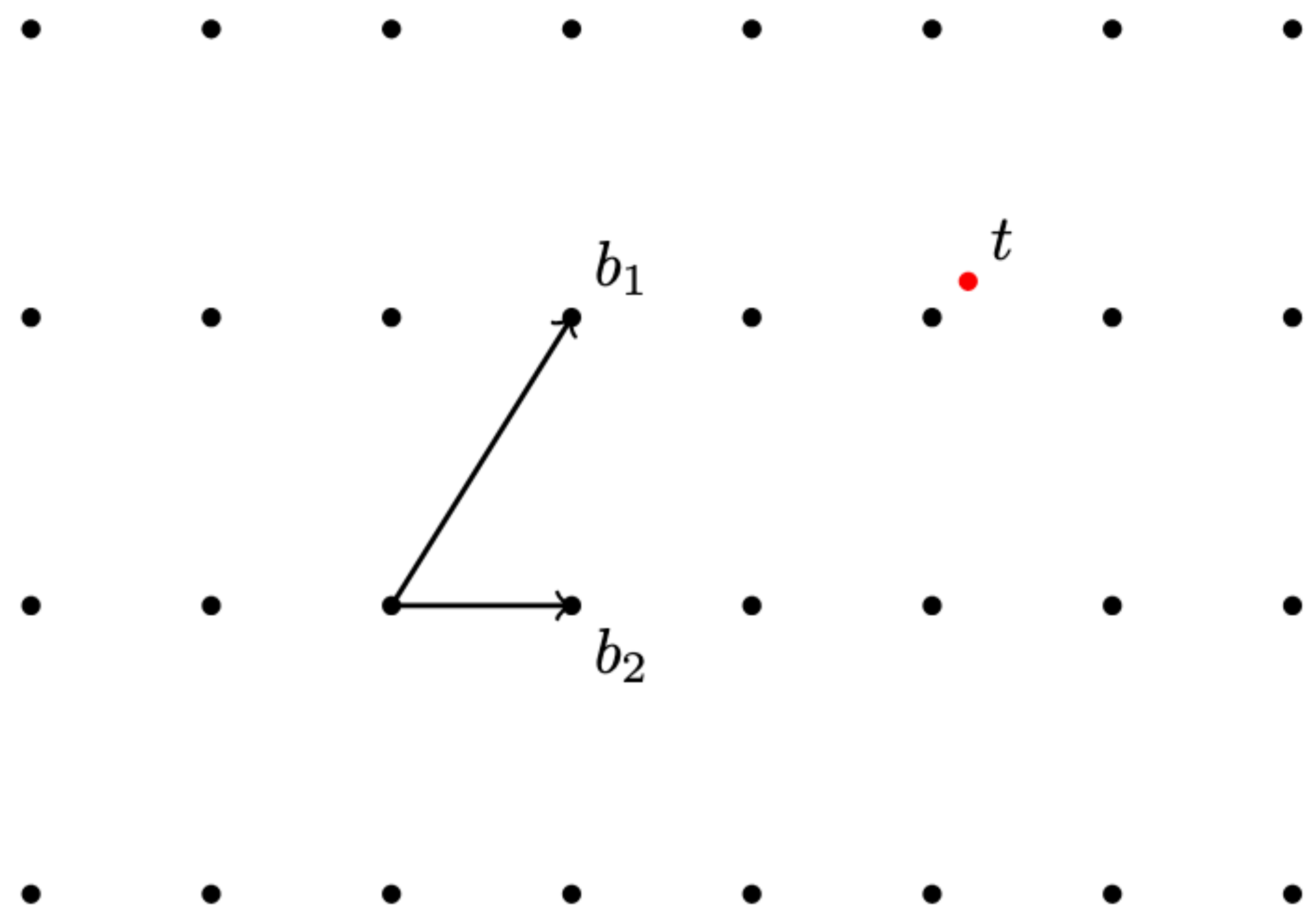
SLR
BDD

BDD_α (Bounded Distance Decoding)

BDD_α (Bounded Distance Decoding)

Given: Lattice basis B and target vector $t = Bz^* + e$

Generated lattice $\mathscr{L}(B) = \{Bz \mid \text{integral } z \in \mathbb{Z}^d\}$

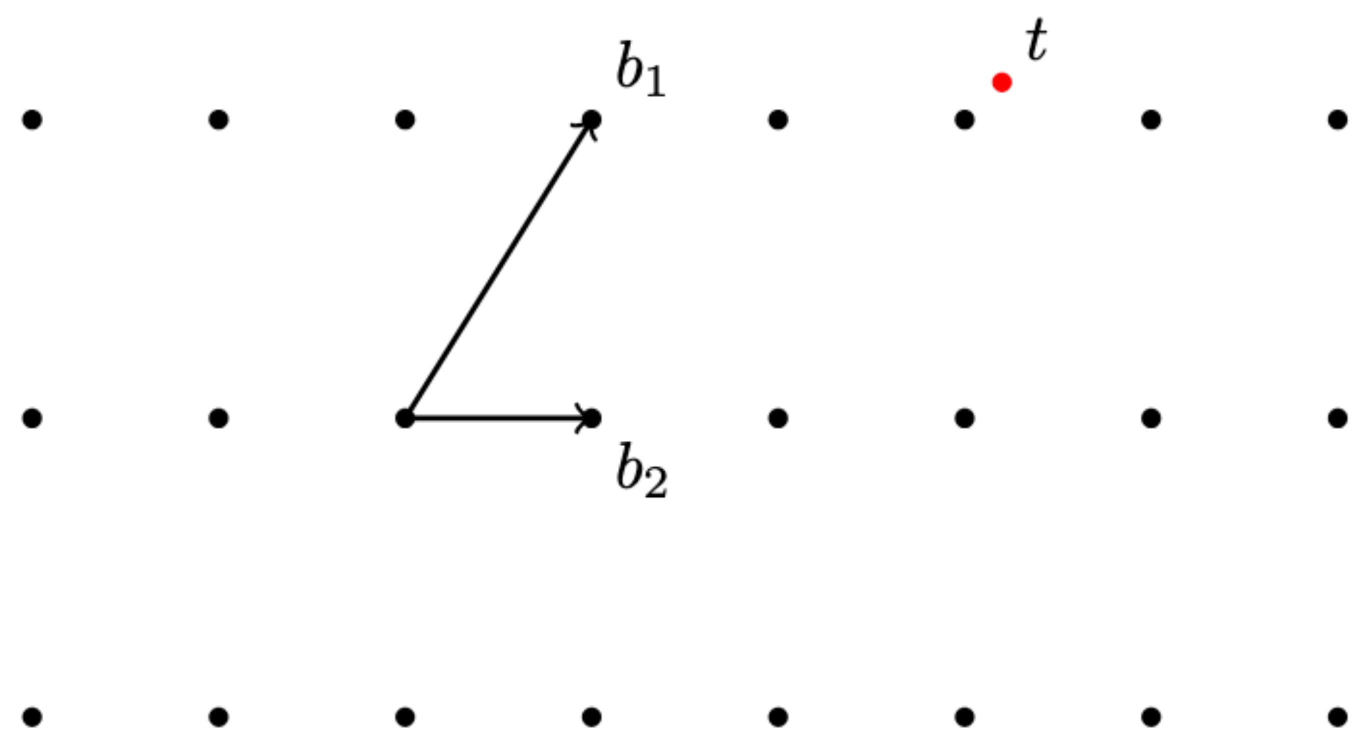


BDD_α (Bounded Distance Decoding)

Given: Lattice basis B and target vector $t = Bz^* + e$

Generated lattice $\mathcal{L}(B) = \{Bz \mid \text{integral } z \in \mathbb{Z}^d\}$

Goal: Find nearest lattice point Bz^*



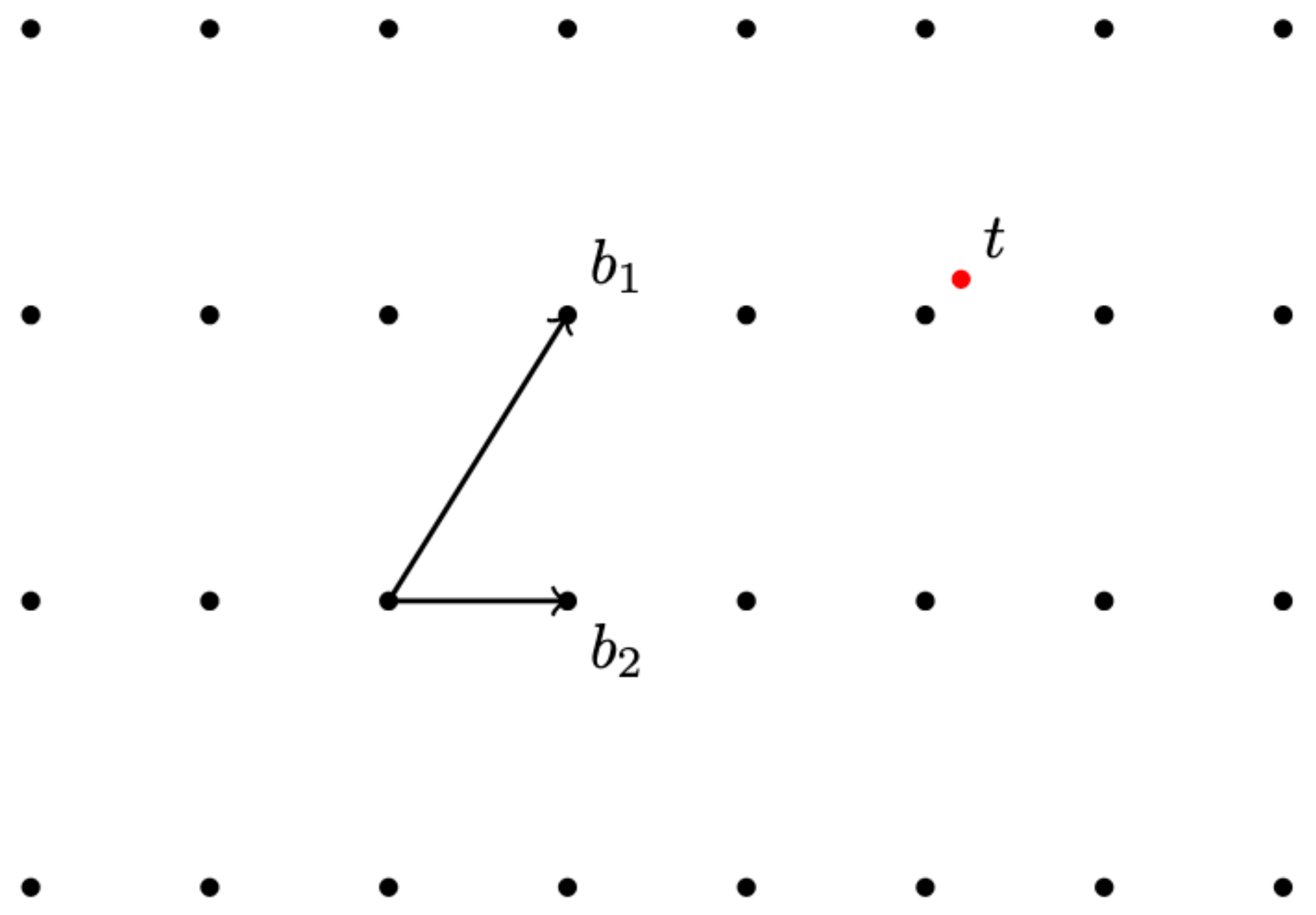
BDD_α (Bounded Distance Decoding)

Given: Lattice basis B and target vector $t = Bz^* + e$

Generated lattice $\mathscr{L}(B) = \{Bz \mid \text{integral } z \in \mathbb{Z}^d\}$

Goal: Find nearest lattice point Bz^*

Promise that $\|e\|_2 \leq \alpha \cdot \lambda_1(B)$
where parameter $\alpha < 1/2$.



BDD_α (Bounded Distance Decoding)

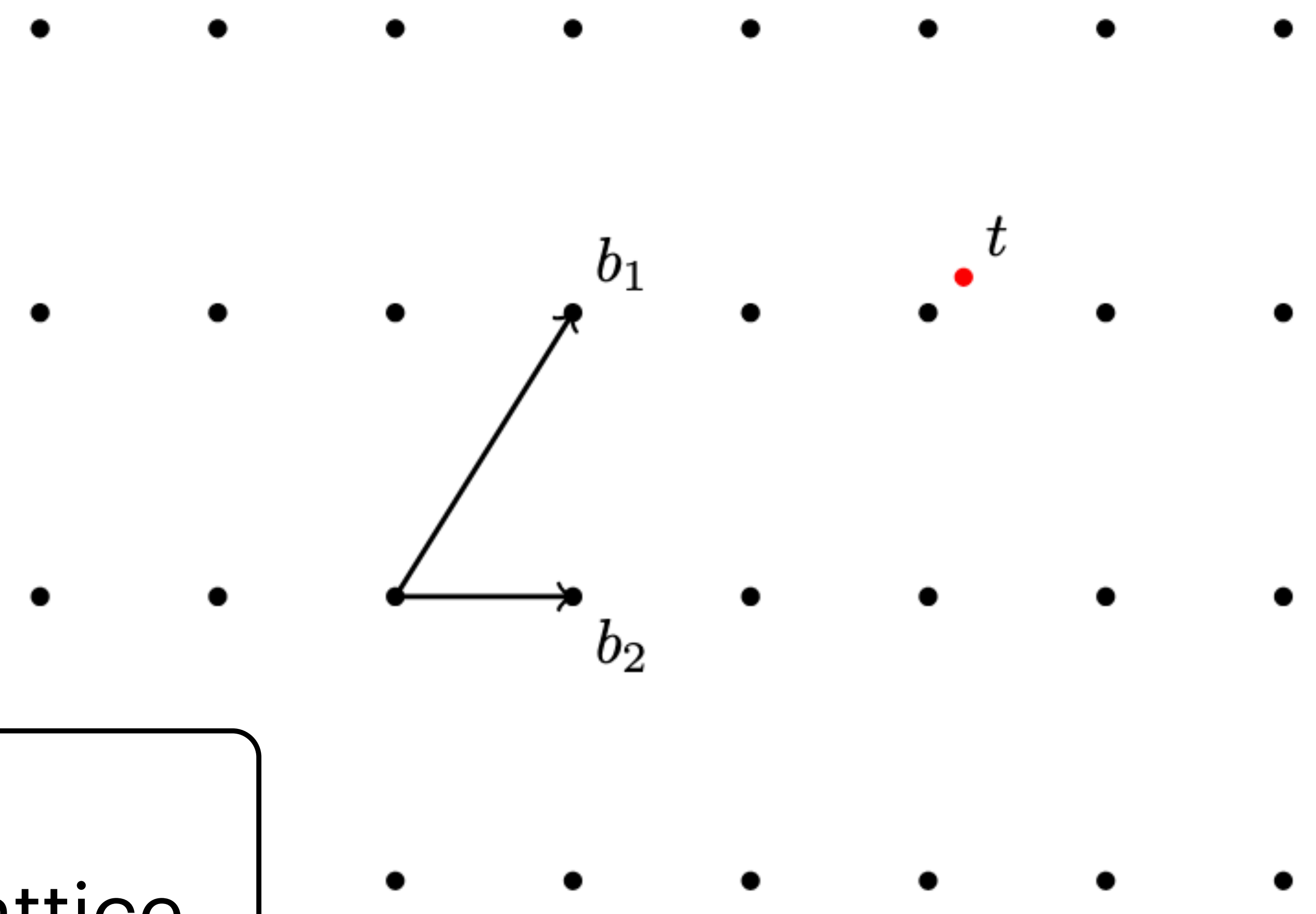
Given: Lattice basis B and target vector $t = Bz^* + e$

Generated lattice $\mathscr{L}(B) = \{Bz \mid \text{integral } z \in \mathbb{Z}^d\}$

Goal: Find nearest lattice point Bz^*

Promise that $\|e\|_2 \leq \alpha \cdot \lambda_1(B)$
where parameter $\alpha < 1/2$.

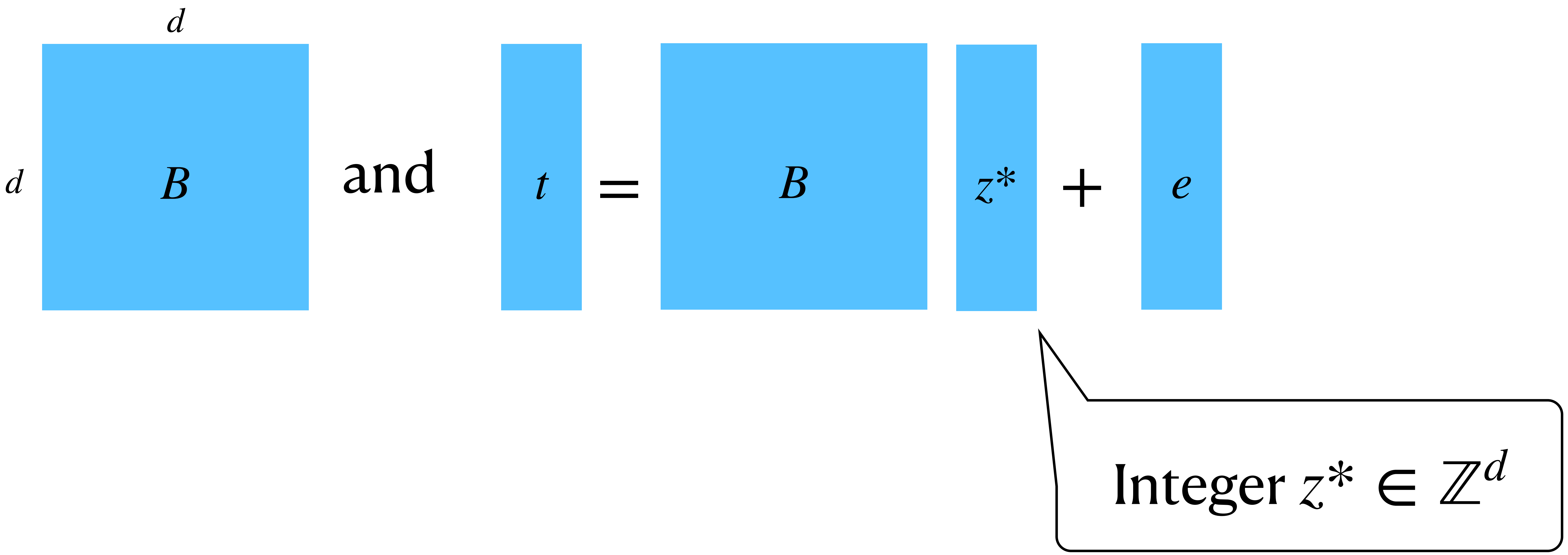
$\lambda_1(B)$:= shortest vector of lattice.
Promise ensures uniqueness.



BDD_α (Bounded Distance Decoding)

Equivalently,

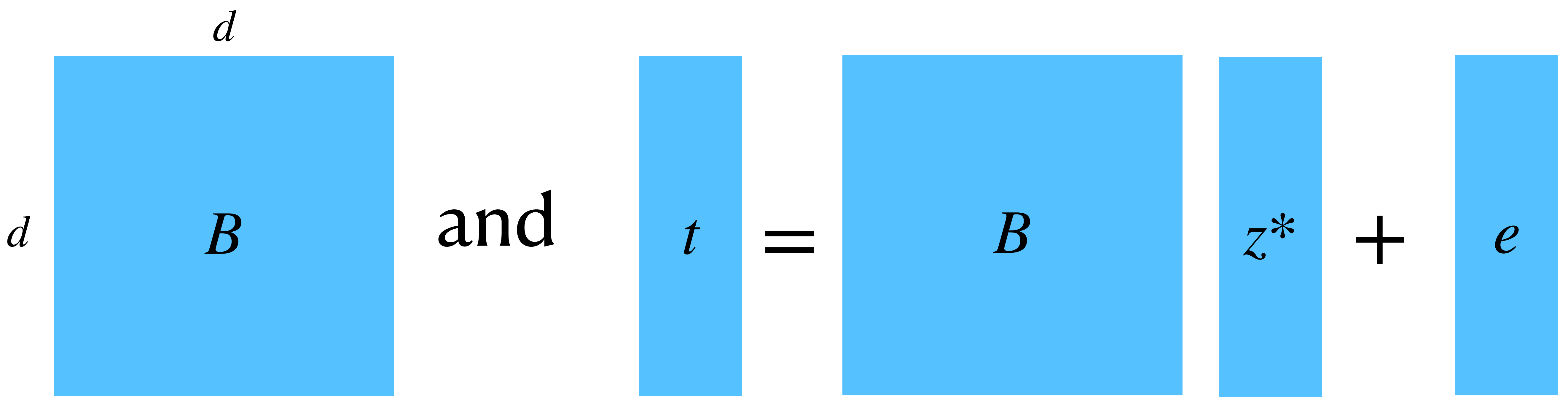
Given:



BDD_α (Bounded Distance Decoding)

Equivalently,

Given:



Integer $z^* \in \mathbb{Z}^d$

Goal: Find $z^* \in \mathbb{Z}^d$

Promised that $\|e\|_2 \leq \alpha \cdot \lambda_1(B)$

BDD_α (Bounded Distance Decoding)

Algorithms	Runtime	Parameter range
[Ajtai-Kumar-Sivakumar '01], [Micciancio-Voulgaris'13], [Aggarwal-Dadush-Stephens-Davidowitz'15]	$2^{d+o(d)}$ 😢	Works for $\alpha < \frac{1}{2}$ ☺️

BDD_α (Bounded Distance Decoding)

Algorithms	Runtime	Parameter range
[Ajtai-Kumar-Sivakumar '01], [Micciancio-Voulgaris'13], [Aggarwal-Dadush-Stephens-Davidowitz'15]	$2^{d+o(d)}$ 😢	Works for $\alpha < \frac{1}{2}$ ☺️
Babai's rounding off algorithm round$(B^{-1}t)$	poly(d) ☺️	Works only for $\alpha \leq \frac{1}{\kappa(B)}$ 😢

BDD_α (Bounded Distance Decoding)

Algorithms	Runtime	Parameter range
[Ajtai-Kumar-Sivakumar '01], [Micciancio-Voulgaris'13], [Aggarwal-Dadush-Stephens-Davidowitz'15]	$2^{d+o(d)}$ 😢	Works for $\alpha < \frac{1}{2}$ ☺️
Babai's rounding off algorithm round$(B^{-1}t)$	poly(d) ☺️	Works only for $\alpha \leq \frac{1}{\kappa(B)}$ 😢

Condition number of lattice basis

$$\kappa(B) = \frac{\sigma_{\max}(B)}{\sigma_{\min}(B)}$$

BDD_α (Bounded Distance Decoding)

Algorithms	Runtime	Parameter range
[Ajtai-Kumar-Sivakumar '01], [Micciancio-Voulgaris'13], [Aggarwal-Dadush-Stephens-Davidowitz'15]	$2^{d+o(d)}$ 😢	Works for $\alpha < \frac{1}{2}$ ☺️
Babai's rounding off algorithm round$(B^{-1}t)$	poly(d) ☺️	Works only for $\alpha \leq \frac{1}{\kappa(B)}$ 😢
LLL + Babai's rounding off algorithm	poly(d) ☺️	Works only for $\alpha \leq \frac{1}{2^{d/2}}$ 😢

SLR (Sparse Linear Regression)	BDD (Bounded Distance Decoding)
Given: X and $y = X\theta^* + w$ **Goal:** Find sparse $\widehat{\theta}$ to minimize $\|X\widehat{\theta} - X\theta^*\|_2$	**Given:** B and $t = Bz^* + e$ **Goal:** Find integer z^*

SLR (Sparse Linear Regression)	BDD (Bounded Distance Decoding)
Given: X and $y = X\theta^* + w$ **Goal:** Find sparse $\widehat{\theta}$ to minimize $\|X\widehat{\theta} - X\theta^*\|_2$	**Given:** B and $t = Bz^* + e$ **Goal:** Find integer z^*

Information-theoretically, the conditioning of the matrix has no bearing on the problem.

<table>
<tr><th>SLR (Sparse Linear Regression)</th><th>BDD (Bounded Distance Decoding)</th></tr>
<tr><td>Given: X and $y = X\theta^* + w$
Goal: Find sparse $\widehat{\theta}$ to minimize
$\|X\widehat{\theta} - X\theta^*\|_2$</td><td>Given: B and $t = Bz^* + e$
Goal: Find integer z^*</td></tr>
<tr><td colspan="2">Information-theoretically, the conditioning of the matrix has no bearing on the problem.</td></tr>
<tr><td colspan="2">Efficient algorithms work for “well-conditioned” matrices:</td></tr>
<tr><td>LASSO: Restricted eigenvalue constant $\mathrm{RE}(X)$</td><td>Babai: Condition number $\kappa(B)$</td></tr>
</table>

<table>
<tr><th>SLR (Sparse Linear Regression)</th><th>BDD (Bounded Distance Decoding)</th></tr>
<tr><td>Given: X and $y = X\theta^* + w$
Goal: Find sparse $\widehat{\theta}$ to minimize $\|X\widehat{\theta} - X\theta^*\|_2$</td><td>Given: B and $t = Bz^* + e$
Goal: Find integer z^*</td></tr>
<tr><td colspan="2">Information-theoretically, the conditioning of the matrix has no bearing on the problem.</td></tr>
<tr><td colspan="2">Efficient algorithms work for “well-conditioned” matrices:</td></tr>
<tr><td>LASSO: Restricted eigenvalue constant $\mathrm{RE}(X)$</td><td>Babai: Condition number $\kappa(B)$</td></tr>
</table>

Is there a more formal connection between SLR and BDD?

Main Result

Theorem 1. Suppose there is a poly-time ***β*-improvement over LASSO** for k-SLR

i.e. achieves prediction error $\varepsilon \leq O\left(\frac{k}{\mathrm{RE}(X)^{1-\beta}} \cdot \|w\|_2\right)$.

Then, there is a poly-time algorithm that solves $\mathrm{BinaryBDD}_\alpha{}^*$ with $\alpha \leq \frac{1}{\mathrm{poly}(d) \cdot \kappa(B)^{2-2\beta}}$.

Beats Babai's dependence on the condition number $\kappa(B)$ if $\beta > 1/2$.

Main Result

Theorem 1. Suppose there is a poly-time ***β*-improvement over LASSO** for k-SLR

i.e. achieves prediction error $\varepsilon \leq O\left(\frac{k}{\mathrm{RE}(X)^{1-\beta}} \cdot \|w\|_2\right)$.

LASSO is $\beta = 0$

Then, there is a poly-time algorithm that solves $\mathrm{BinaryBDD}_\alpha^*$ with $\alpha \leq \frac{1}{\mathrm{poly}(d) \cdot \kappa(B)^{2-2\beta}}$.

Beats Babai's dependence on the condition number $\kappa(B)$ if $\beta > 1/2$.

Main Result

Theorem 1. Suppose there is a poly-time **β-improvement over LASSO** for k-SLR

i.e. achieves prediction error $\varepsilon \leq O\left(\frac{k}{\mathrm{RE}(X)^{1-\beta}} \cdot \|w\|_2\right)$.

LASSO is $\beta = 0$

Then, there is a poly-time algorithm that solves $\mathrm{BinaryBDD}_{\alpha}$* with $\alpha \leq \frac{1}{\mathrm{poly}(d) \cdot \kappa(B)^{2-2\beta}}$.

Beats Babai's dependence on the condition number $\kappa(B)$ if $\beta > 1/2$.

* BinaryBDD: variant of BDD where we restrict $z^* \in \{\pm 1\}^d$ instead of $\mathbb{Z}^d$.
As far as we know, BinaryBDD's hardness profile is similar to that of BDD [Kirchner-Fouque'25].

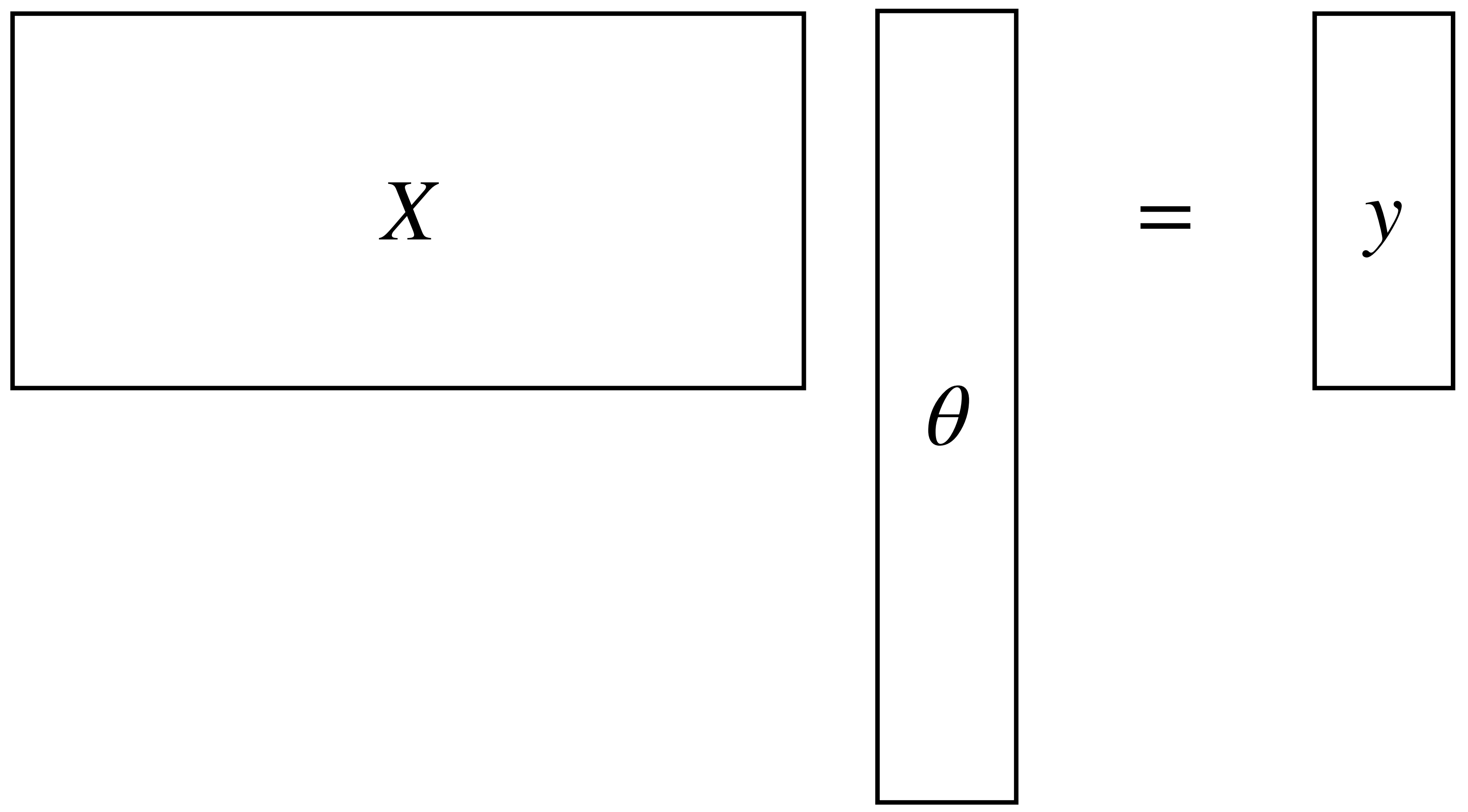
Main Result

Simple gadget that transforms a **sparsity** constraint into an **integrality** constraint.

Main Result

Simple gadget that transforms a **sparsity** constraint into an **integrality** constraint.

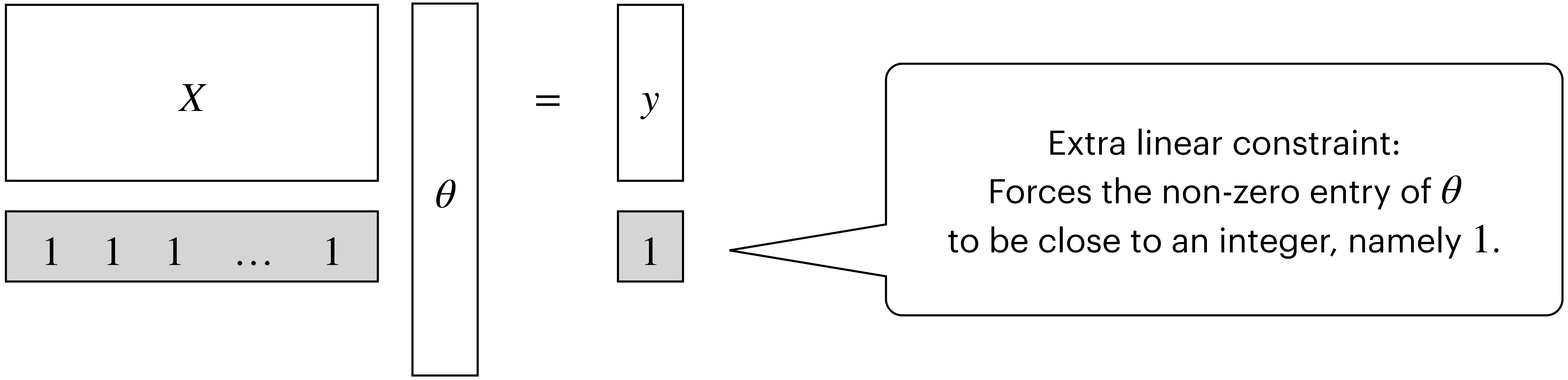
Toy example: Sparsity $k = 1$.



Main Result

Simple gadget that transforms a **sparsity** constraint into an **integrality** constraint.

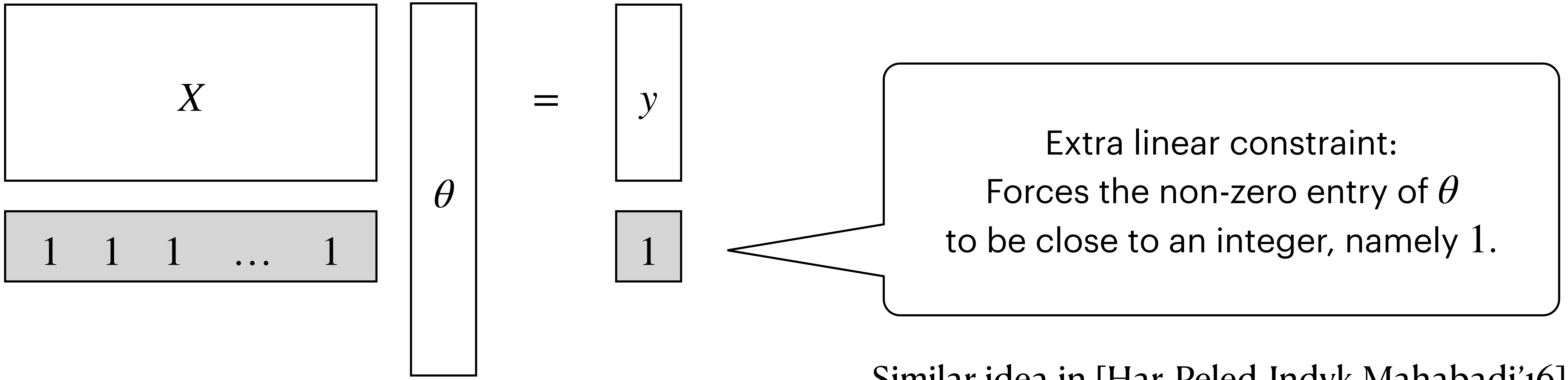
Toy example: Sparsity $k = 1$.



Main Result

Simple gadget that transforms a **sparsity** constraint into an **integrality** constraint.

Toy example: Sparsity $k = 1$.



Similar idea in [Har-Peled-Indyk-Mahabadi'16]

Main Result

Our hard k-SLR instances have design matrices $X = \begin{pmatrix} X_1 \\ G_2 \end{pmatrix}$, where

- Gaussian $X_1 \sim \mathcal{N}(0, \Sigma)^m$ where $\Sigma = G_1^\top B^\top B G_1$
    - Covariance matrix Σ is closely related to lattice basis B
- Gadget matrices G_1, G_2 are fixed and independent of instance

Main Result

Our hard k-SLR instances have design matrices $X = \begin{pmatrix} X_1 \\ G_2 \end{pmatrix}$, where

- Gaussian $X_1 \sim \mathcal{N}(0, \Sigma)^m$ where $\Sigma = G_1^\top B^\top B G_1$
  - Covariance matrix Σ is closely related to lattice basis B
- Gadget matrices G_1, G_2 are fixed and independent of instance

Corollary 1. Theorem 1 + lattice-based worst-case to average-case reductions

→ A distribution over design matrices* for which SLR is hard

*Caveat: These design matrices have $\mathrm{RE}(X) = 0$.

Second Result

Theorem 2. In the total regime, where there are many sparse solutions, it is hard to find *any* solution, even for nice design matrices $X = \begin{pmatrix} X_1 \\ G_2 \end{pmatrix}$ where

- The rows of X_1 are i.i.d. standard Gaussian
- G_2 is fixed gadget matrix,

assuming the worst-case hardness of lattice problems.

Proof goes through Continuous Learning With Errors [Bruna-Regev-Song-Tang'21]

Summary

- **Theorem 1.** Unique-solution regime: Hardness of BinaryBDD for a lattice basis B $\implies$ hardness of (essentially*) Gaussian design SLR with covariance related to B.
- **Corollary 1.** First average-case hardness of SLR for Gaussian design matrices*.

* modulo the gadget matrix that enforces the integrality constraint.

Summary

- **Theorem 1.** Unique-solution regime: Hardness of BinaryBDD for a lattice basis B $\implies$ hardness of (essentially*) Gaussian design SLR with covariance related to B.
- **Corollary 1.** First average-case hardness of SLR for Gaussian design matrices*.
- **Theorem 2.** Total regime: hard to find *any* solution for standard Gaussian design matrices*, assuming worst-case hardness of lattices problems.

* modulo the gadget matrix that enforces the integrality constraint.

Open Problem #1

- Kelner-Koehler-Meka-Rohatgi preconditioned LASSO beats the RE bound for some design matrices X.
- What is the relationship between **preconditioned LASSO for SLR [KKMR]** and **basis-reduction in the lattice world?**
- Can techniques from one world improve algorithms in the other?

Open Problem #2

Is there a natural distribution of design matrices for which the RE bound is tight?