Fast and Accurate Least-Mean-Squares Solvers

Alaa Maalouf

Ibrahim Jubran

Dan Feldman

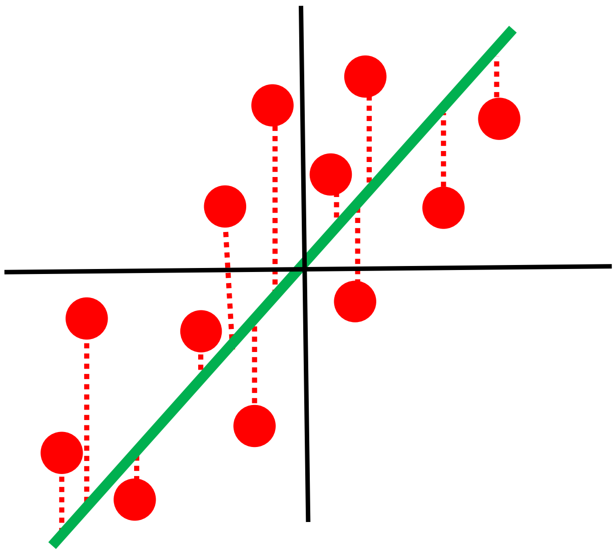
The Robotics & Big Data Lab,
Department of Computer Science,
University of Haifa,
Israel

אוניברסיטת חיפה
University of Haifa
جامعة حيفا

Least-Mean-Squares Solvers

Input: A matrix

$$A = \begin{pmatrix} | & & | \\ a_1 & \ldots & a_n \\ | & & | \end{pmatrix}^T \in \mathbb{R}^{n \times d}$$



and a vector

$$b = (b_1, \cdots, b_n)^T \in \mathbb{R}^n$$

Output: $x^* \in \mathbb{R}^d$

that minimizes

$$\boxed{f(\|Ax - b\|_2^2) + g(x)}$$

constraint

over every $x \in \mathbb{R}^d$, where $f: \mathbb{R} \to \mathbb{R}$ and $g: \mathbb{R}^d \to \mathbb{R}$.

Least-Mean-Squares Solvers

Input: A matrix

$a_n \quad)^T \in \mathbb{R}^{n\times d}$

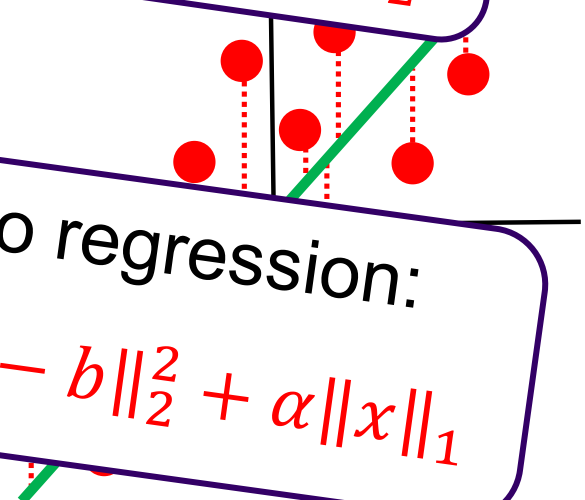
and a vector

$b = (b_1, \cdots, b_n)^T$

$\mathbb{R}^d$

constraint

over every $x \in \mathbb{R}^d$, where f : ... $\mathbb{R}^d \to \mathbb{R}$.



Linear regression:
$\|Ax - b\|_2^2$

Ridge regression:
$\|Ax - b\|_2^2 + \alpha\|x\|_2^2$

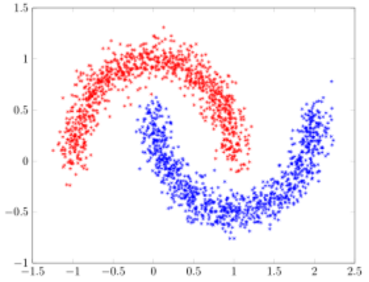
Lasso regression:
$\frac{1}{2n}\|Ax - b\|_2^2 + \alpha\|x\|_1$

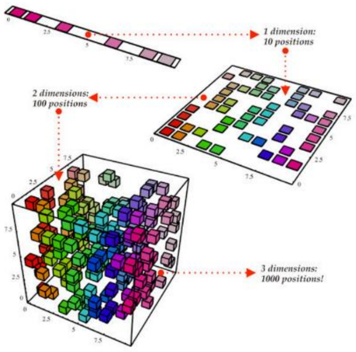
Elastic-net regression:
$\|Ax - b\|_2^2 + \rho\alpha\|x\|_2^2 + \frac{(1-\rho)}{2}\alpha\|x\|_1$

Applications

- Data analysis
- Prediction
- Feature selection
- Spectral clustering
- Dimensionality reduction





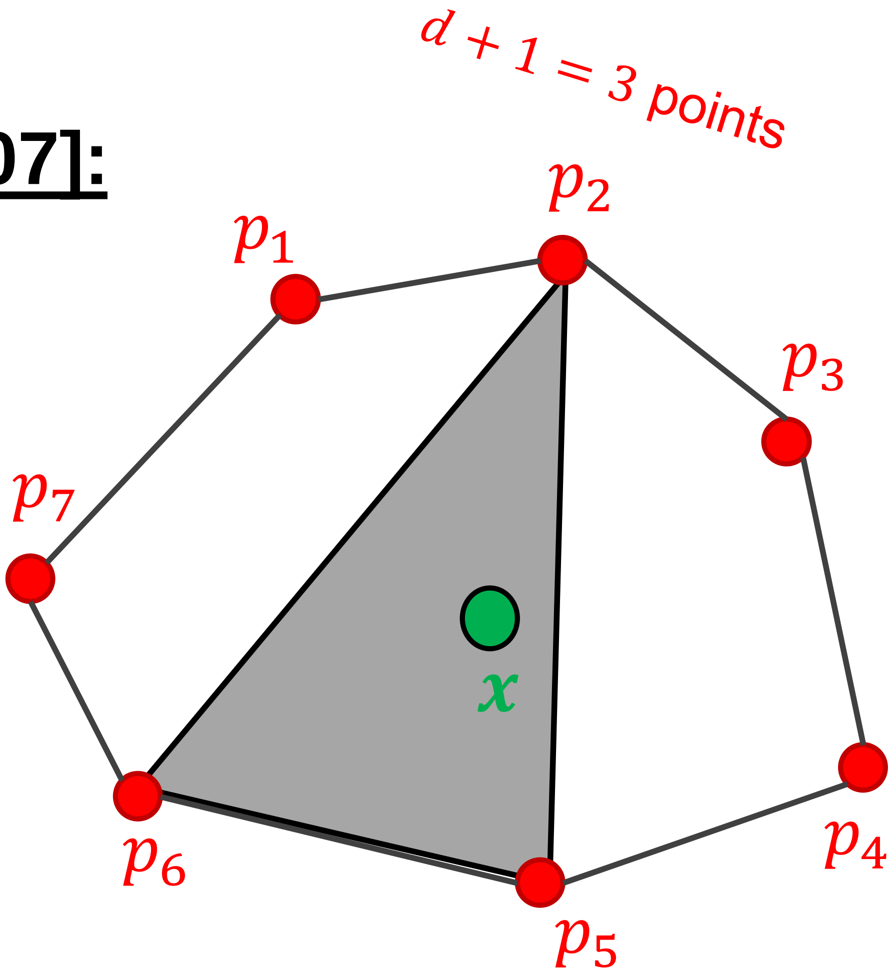


Practical Considerations

- Computation time - The use of cross validations (CV) with a huge number of hyperparameter is time consuming.
- Space complexity - The use of SVD or other factorizations on massive input leads to extensive memory usage.
- Numerical stability - There are faster yet numerically unstable solutions.

Main Technique

Theorem [Caratheodory 1907]:



Main Technique

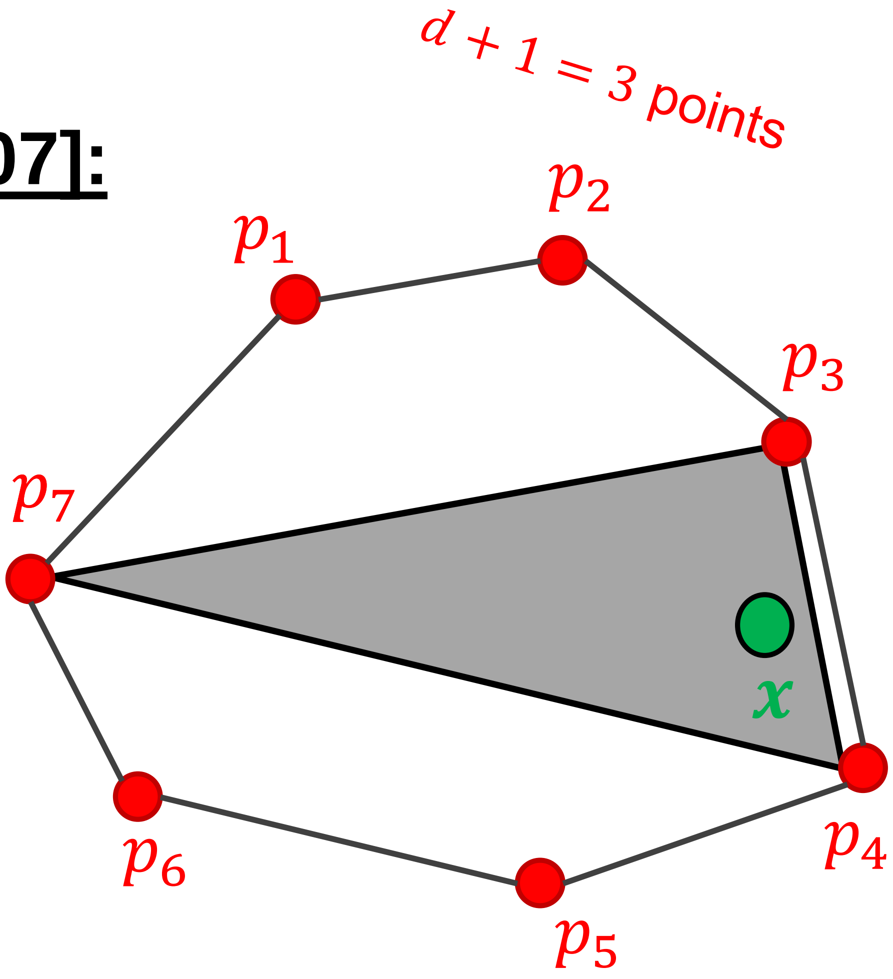
<u>Theorem [Caratheodory 1907]:</u>

This *<u>Caratheodory set</u>* can be computed in $\boxed{O(n^2d^2)\text{ time.}}$

Bottleneck

Goal

$\boxed{\boldsymbol{O(nd)}}$



Our Contribution

- Computing the *Caratheodory set* in $O(nd + \log(n)\, d^4)$ time.

- Practically and provably boosting:
  - time complexity of LMS solvers.
  - space complexity of LMS solvers.
  - numerical accuracy of LMS solvers.

- Evaluation on real-world data.

- Full open source code.

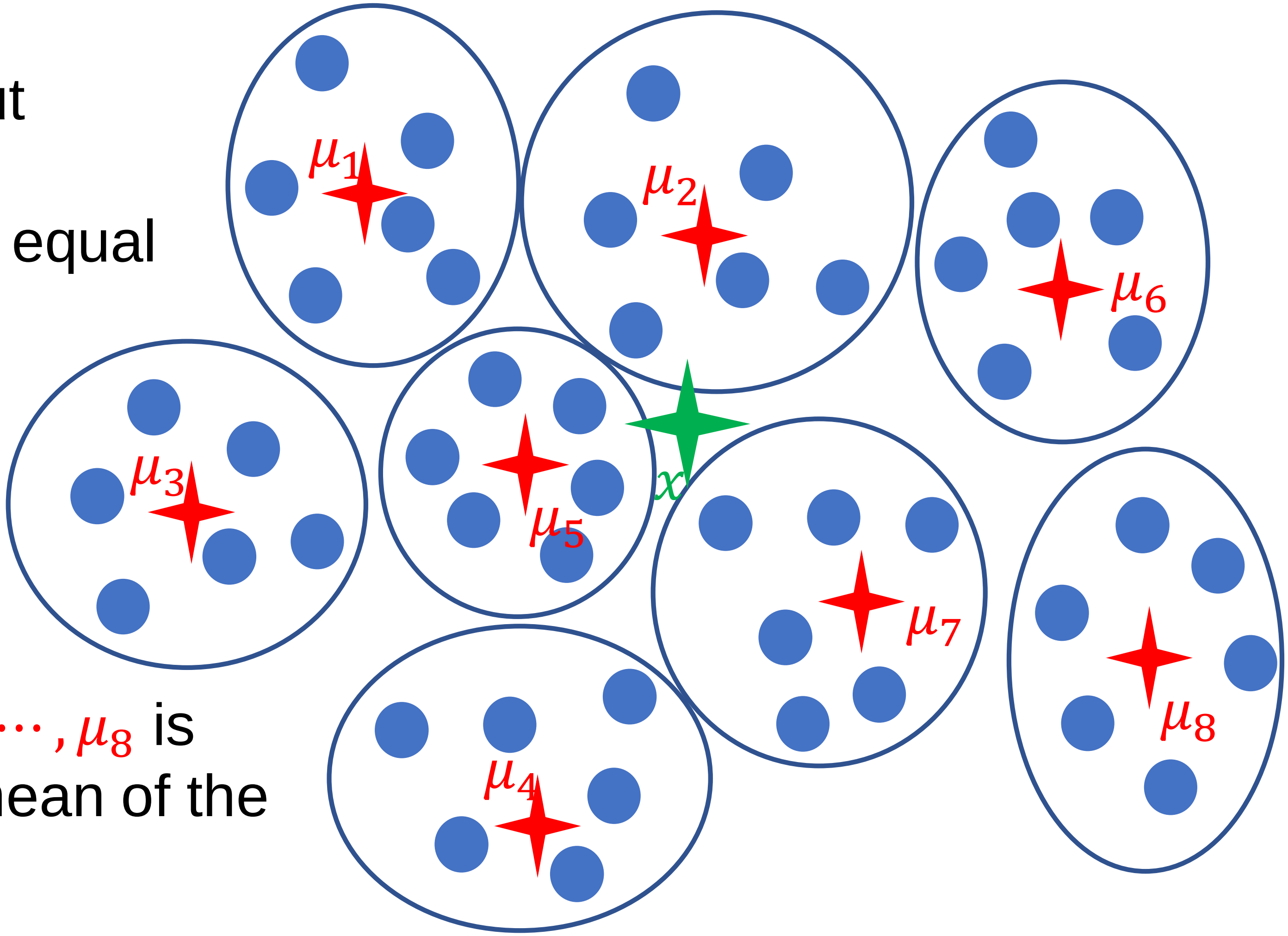
Caratheodory Booster Illustration

Step 1:

Partition the input d-dimensional points into $k = 8$ equal sized subsets.

$\mu_1, \cdots, \mu_8$ are the subset means.

The mean of $\mu_1, \cdots, \mu_8$ is equal to x, the mean of the input.

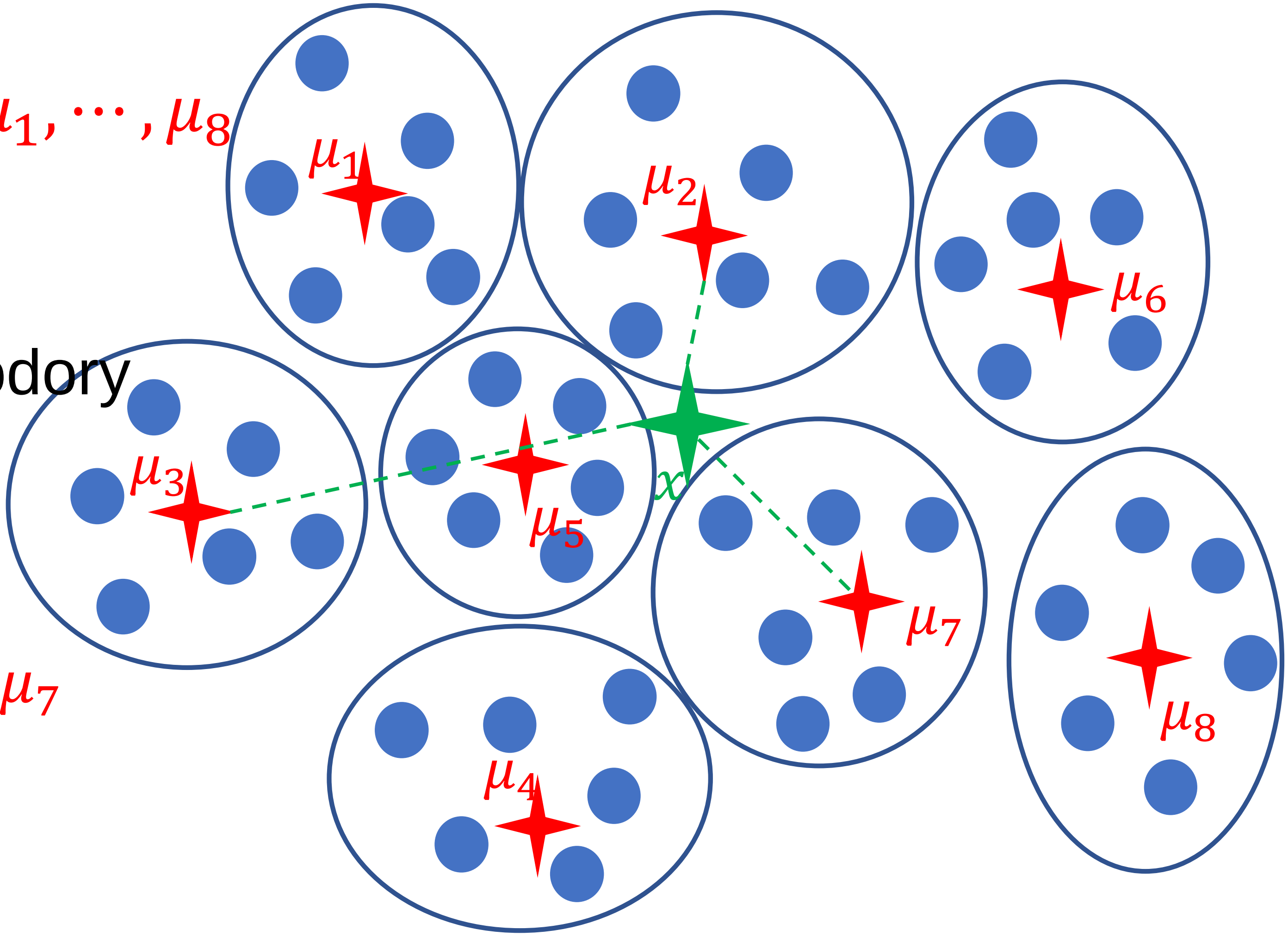


Caratheodory Booster Illustration

Step 2:

Consider only $\mu_1, \cdots, \mu_8$ and x.

Apply Caratheodory to represent x as a convex combination of a subset μ_2, μ_3, μ_7 of the means.

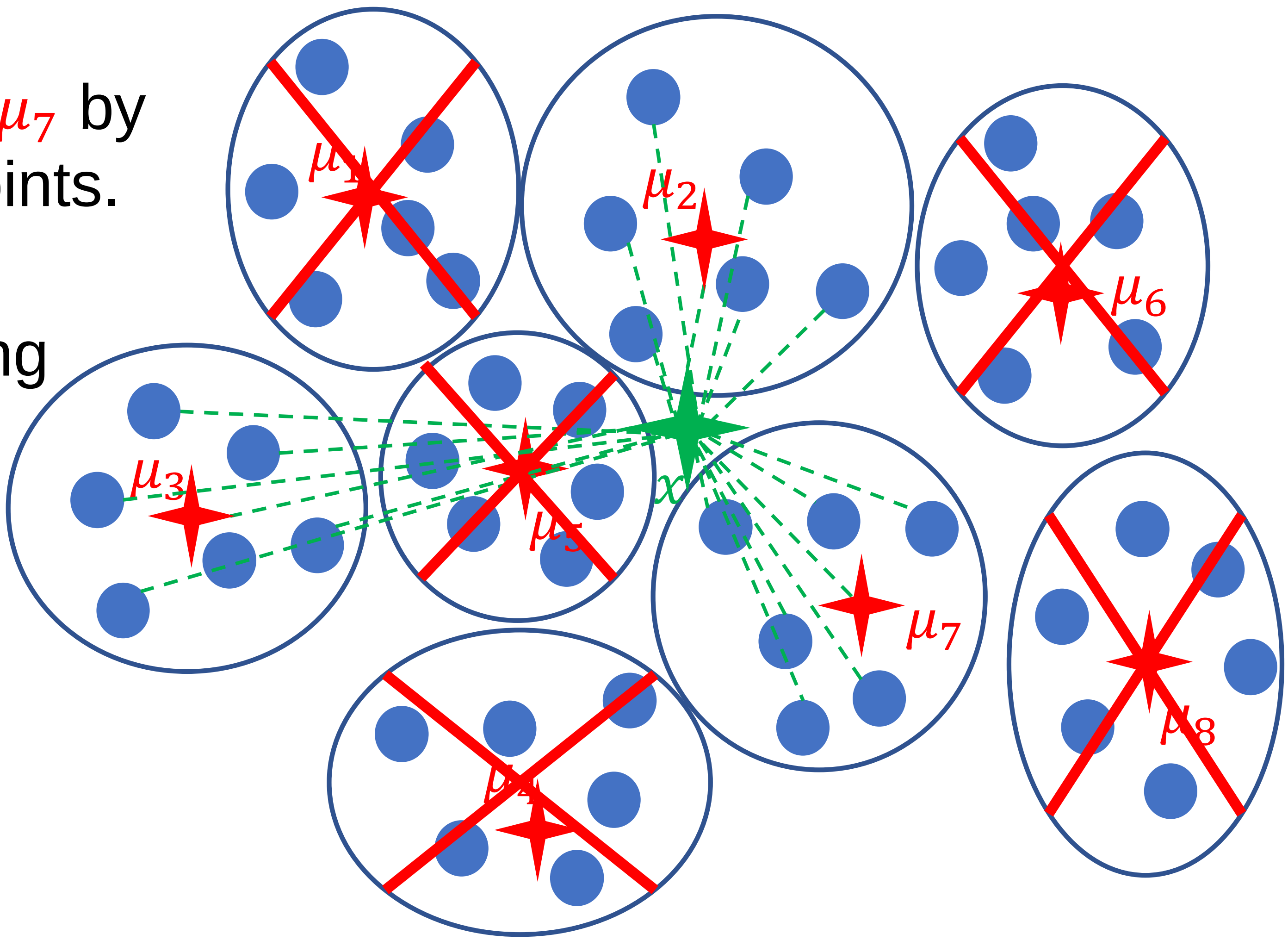


Caratheodory Booster Illustration

Step 3:

Replace μ_2, μ_3, μ_7 by their original points.

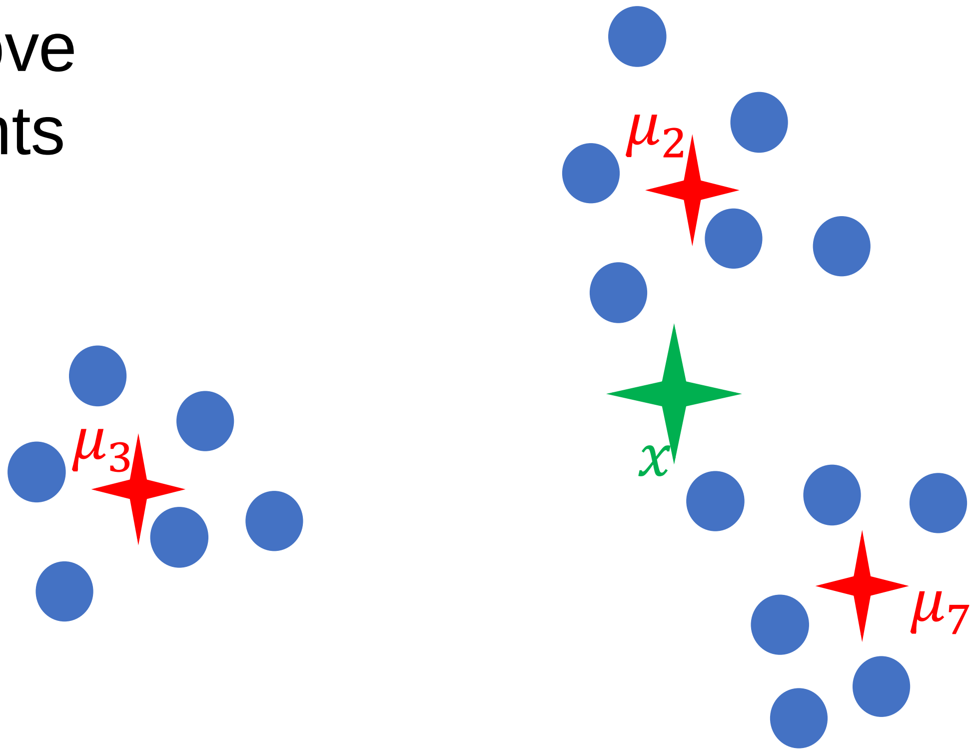
Delete remaining points.



Caratheodory Booster Illustration

Step 4:

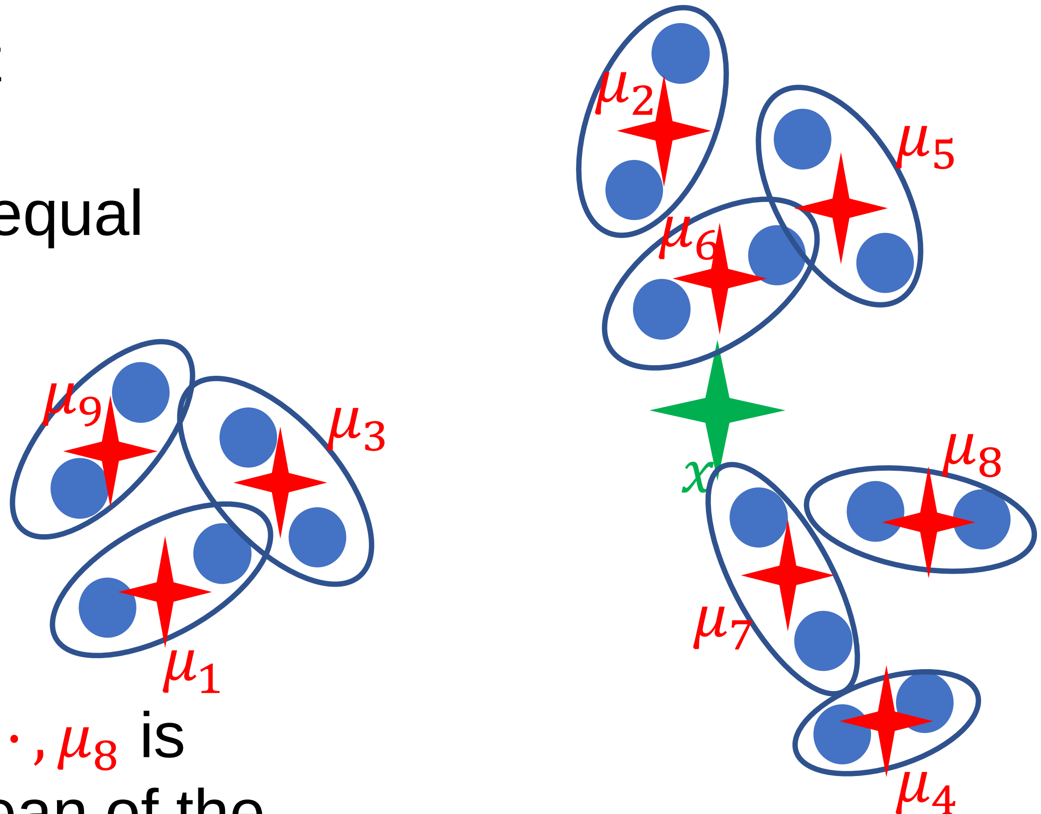
Repeat steps above until only few points remain.



Caratheodory Booster Illustration

<u>Step 1:</u>

Partition the input d-dimensional points into $k = 8$ equal sized subsets.



$\mu_1, \cdots, \mu_8$ are the subset means.

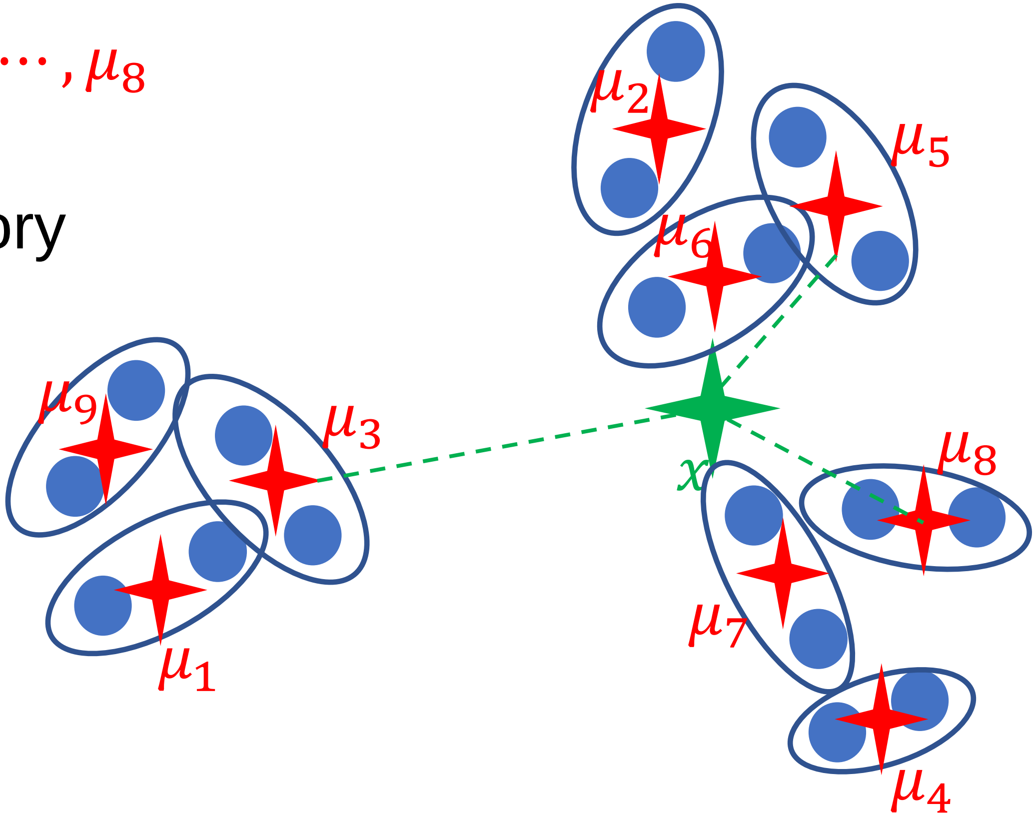
The mean of $\mu_1, \cdots, \mu_8$ is equal to x, the mean of the input.

Caratheodory Booster Illustration

Step 2:

Consider only $\mu_1, \cdots, \mu_8$ and x.

Apply Caratheodory to represent x as a convex combination of a subset μ_2, μ_3, μ_7 of the means.

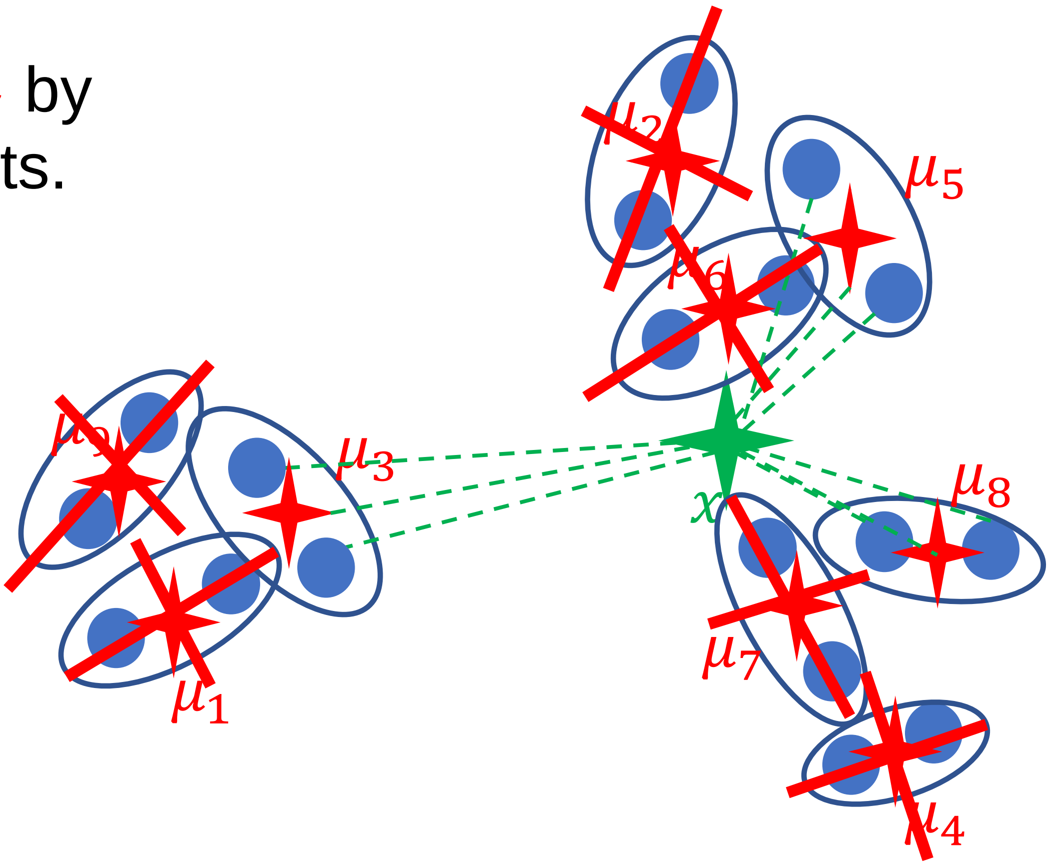


Caratheodory Booster Illustration

Step 3:

Replace μ_2, μ_3, μ_7 by their original points.

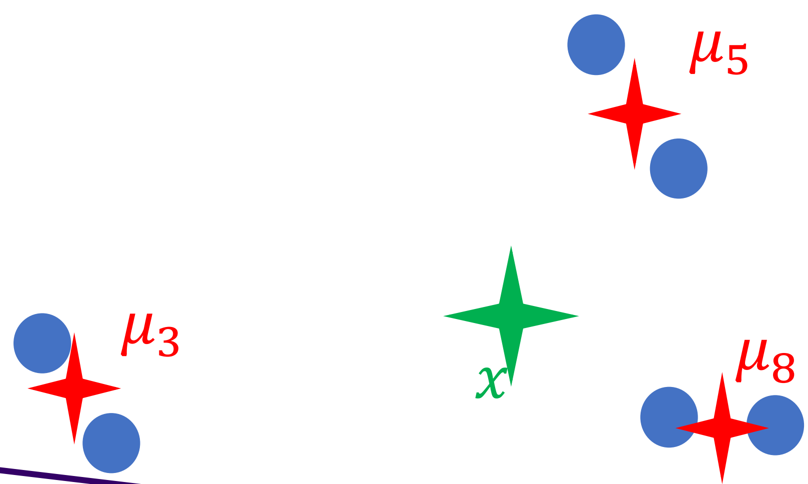
Delete remaining points.



Caratheodory Booster Illustration

Step 4:

Repeat steps above until only few points remain.



$O(nd + d^4 \log n)$, i.e., $O(nd)$ time for sufficiently large n.

Key Observation

$$\|Ax-b\|_2^2 = \left\| [A \mid b] \overbrace{\begin{pmatrix} x \\ -1 \end{pmatrix}}^{x'} \right\|_2^2$$

$$= \|[A \mid b]x'\|_2^2$$

$$= x'^T \boxed{[A \mid b]^T [A \mid b]} x'$$

Covariance matrix

Therefore, for any $m \geq 1, C \in \mathbb{R}^{m \times d}$ and $y \in \mathbb{R}^m$ such that

$$\boxed{\underbrace{[A \mid b]^T [A \mid b]}_{P^T P} = \underbrace{[C \mid y]^T [C \mid y]}_{Z^T Z},}$$

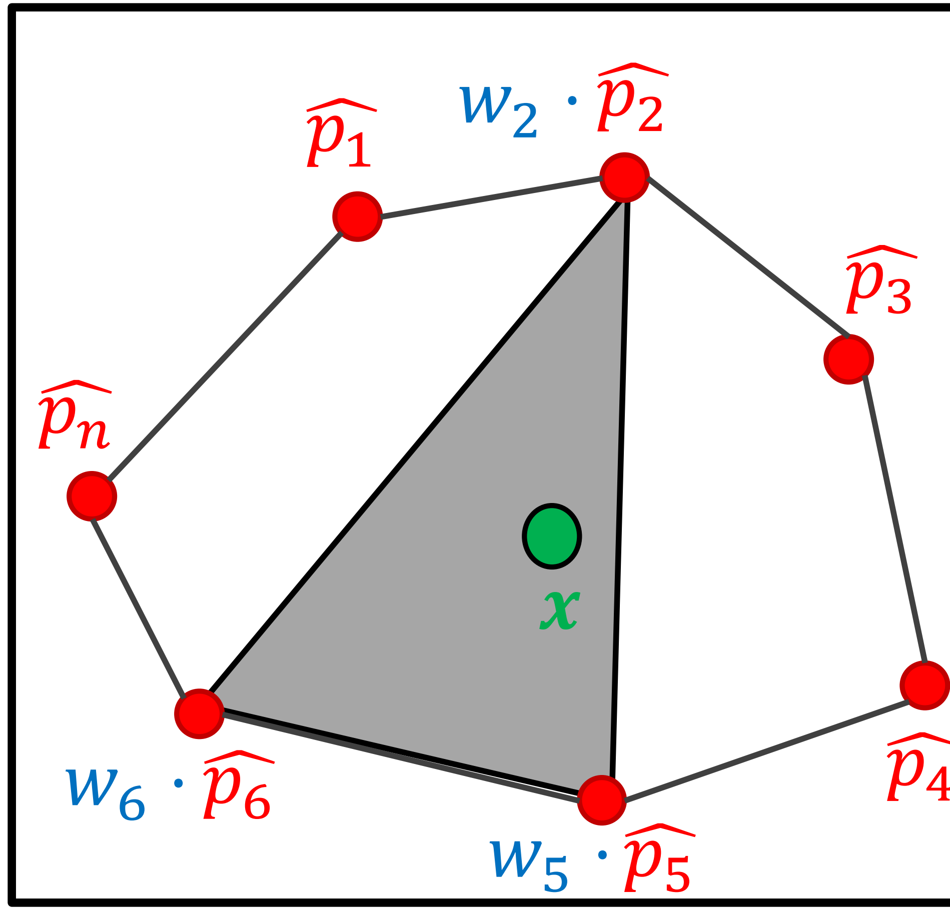
we have that:

$$f(\|Ax-b\|_2^2) + g(x) = f\left(x'^T P^T P x'\right) + g(x)$$

$$= f\left(x'^T Z^T Z x'\right) + g(x)$$

$$= f(\|Cx-y\|_2^2) + g(x)$$

Maintaining the Covariance Matrix



Caratheodory

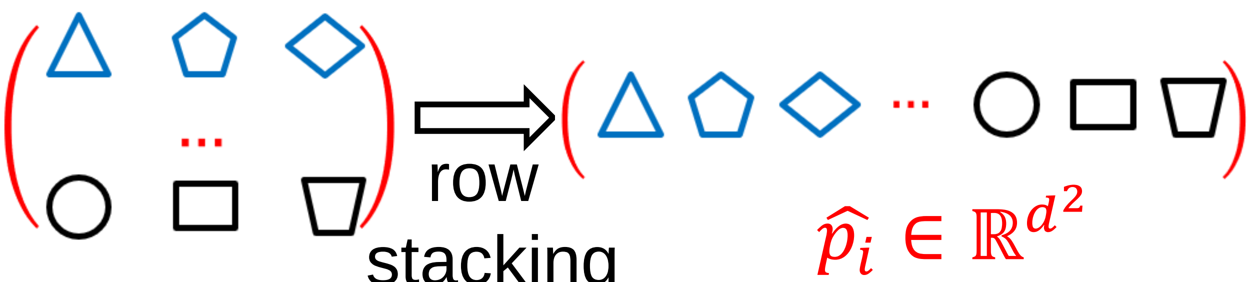
Input:

$$P = \begin{pmatrix} | & & | \\ p_1 & \dots & p_n \\ | & & | \end{pmatrix}^T \in \mathbb{R}^{n \times d}$$

Immediate:

$$x = \frac{P^T P}{n} = \frac{1}{n}\sum_i p_i p_i^T$$

$$\in \mathbb{R}^{d \times d}$$



$$\widehat{p_i} \in \mathbb{R}^{d^2}$$

Maintaining the Covariance Matrix

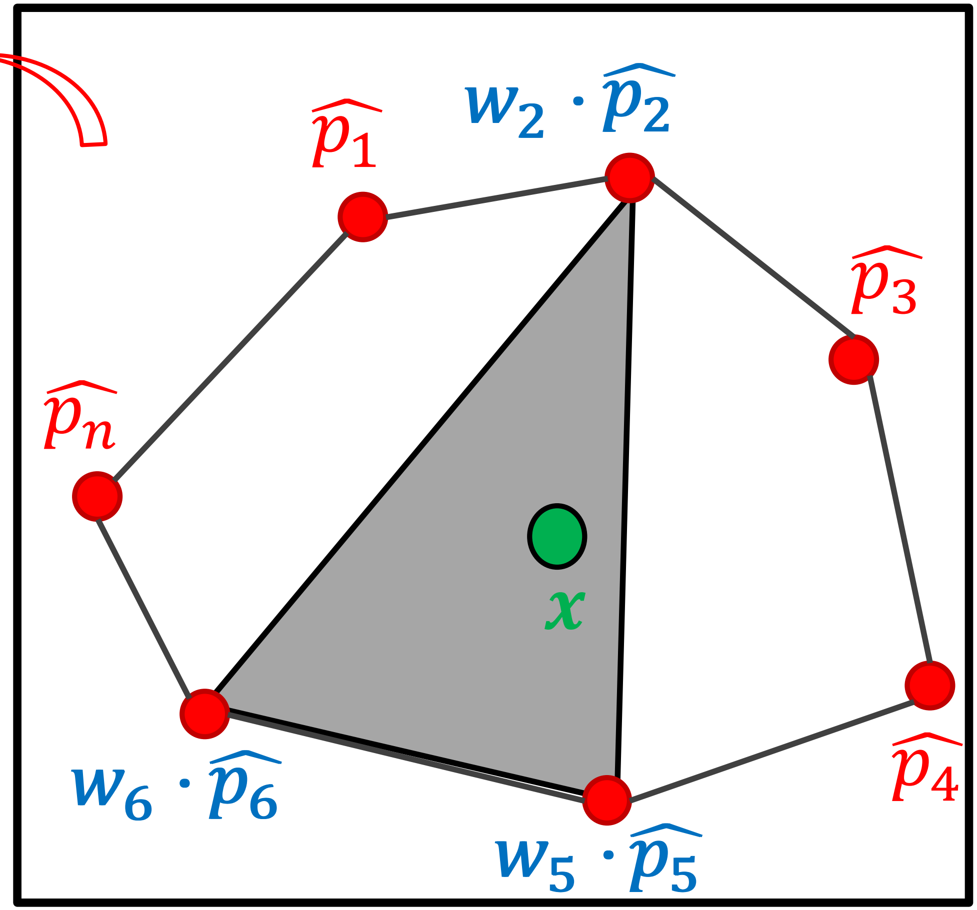
$$\underbrace{\begin{bmatrix}\sqrt{n\cdot w_2}\,p_2^T\\ \sqrt{n\cdot w_5}\,p_5^T\\ \sqrt{n\cdot w_6}\,p_6^T\end{bmatrix}^T}\ \underbrace{\begin{bmatrix}\sqrt{n\cdot w_2}\,p_2^T\\ \sqrt{n\cdot w_5}\,p_5^T\\ \sqrt{n\cdot w_6}\,p_6^T\end{bmatrix}}$$

$$= Z^TZ$$

$$\Downarrow$$

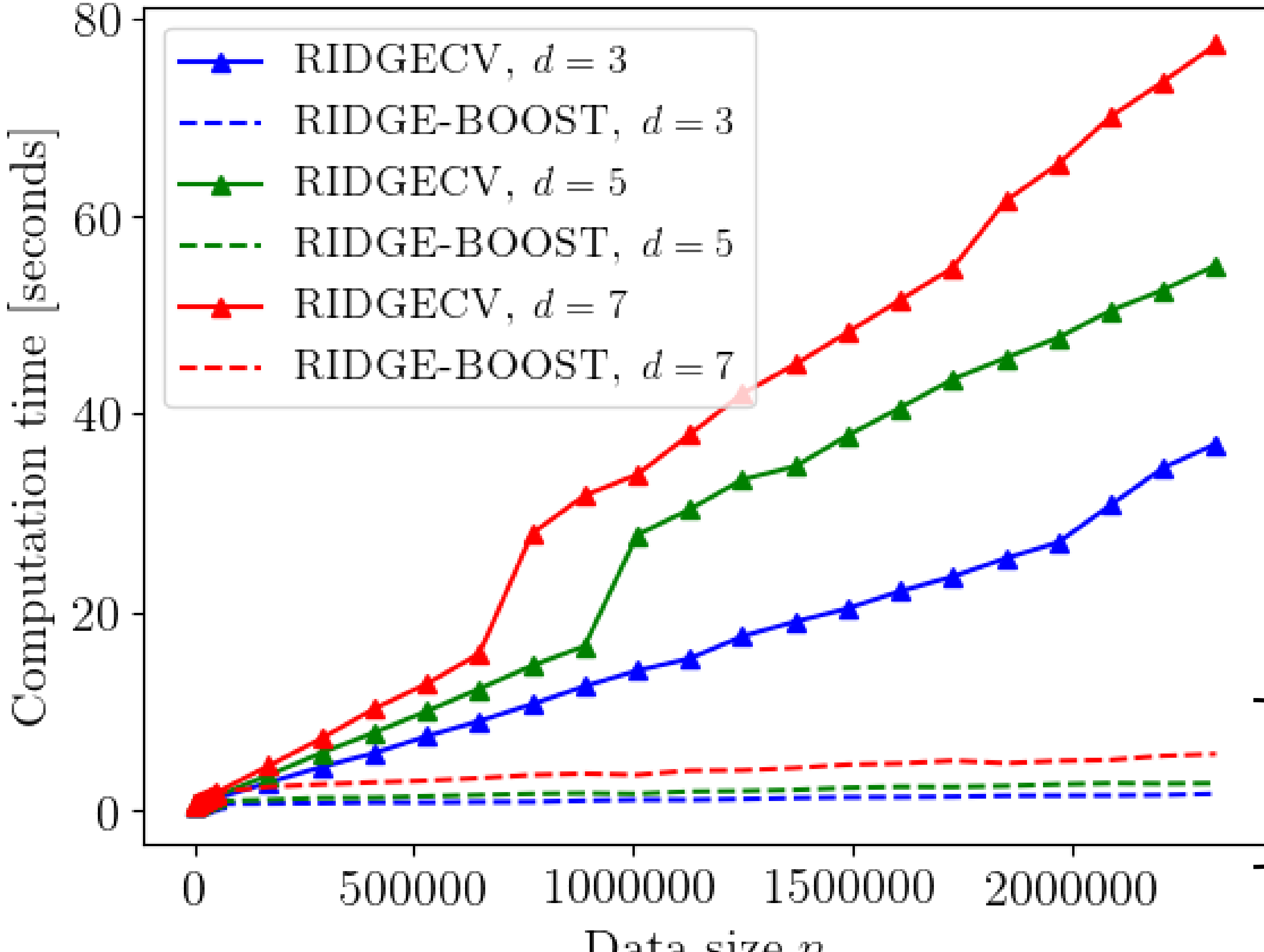
$$\boxed{P^TP = Z^TZ}$$

Preserve covariance accurately!



Caratheodory

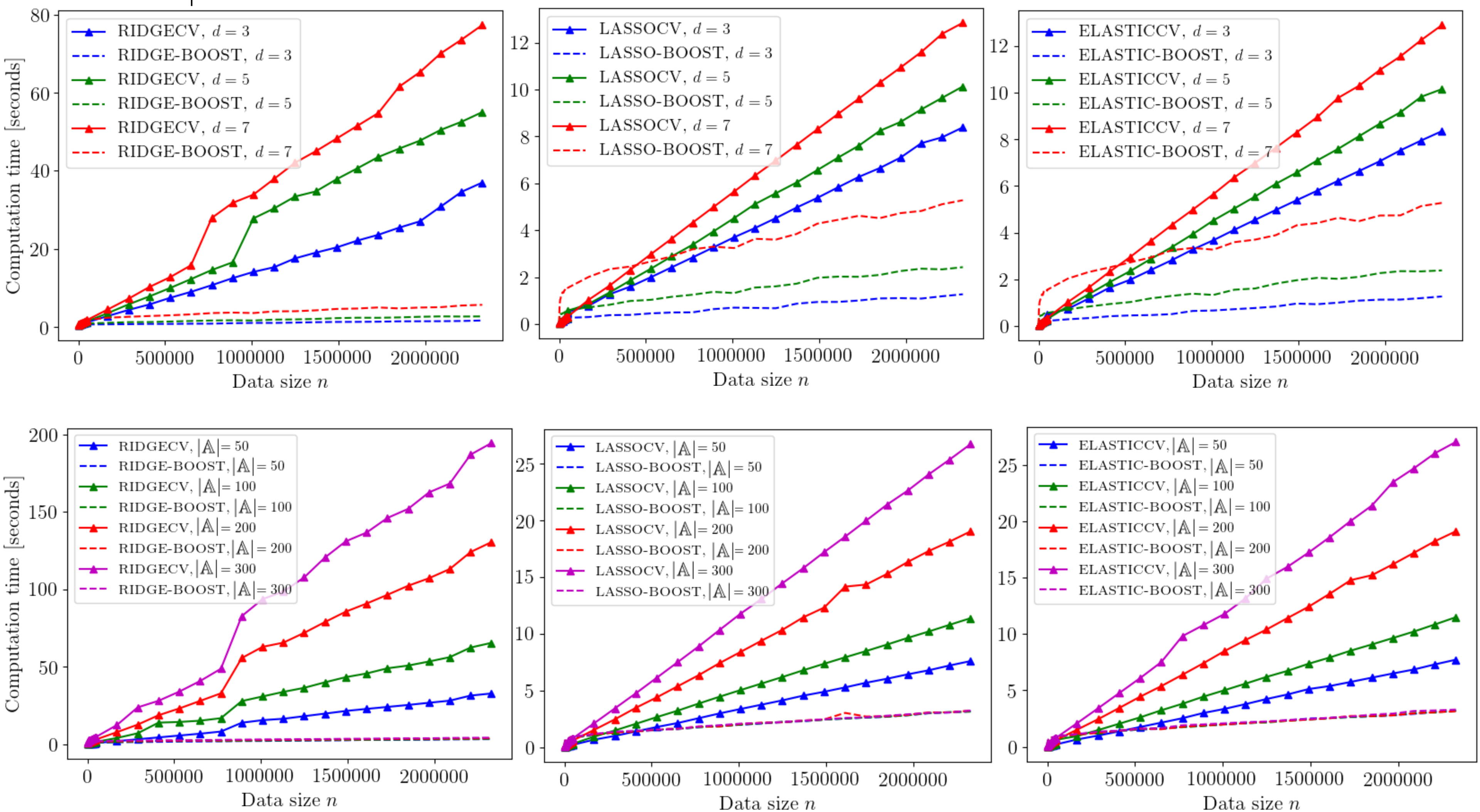
Boosting Computation Time On Real-World Data (roughly x50)



LMS on the **original data**

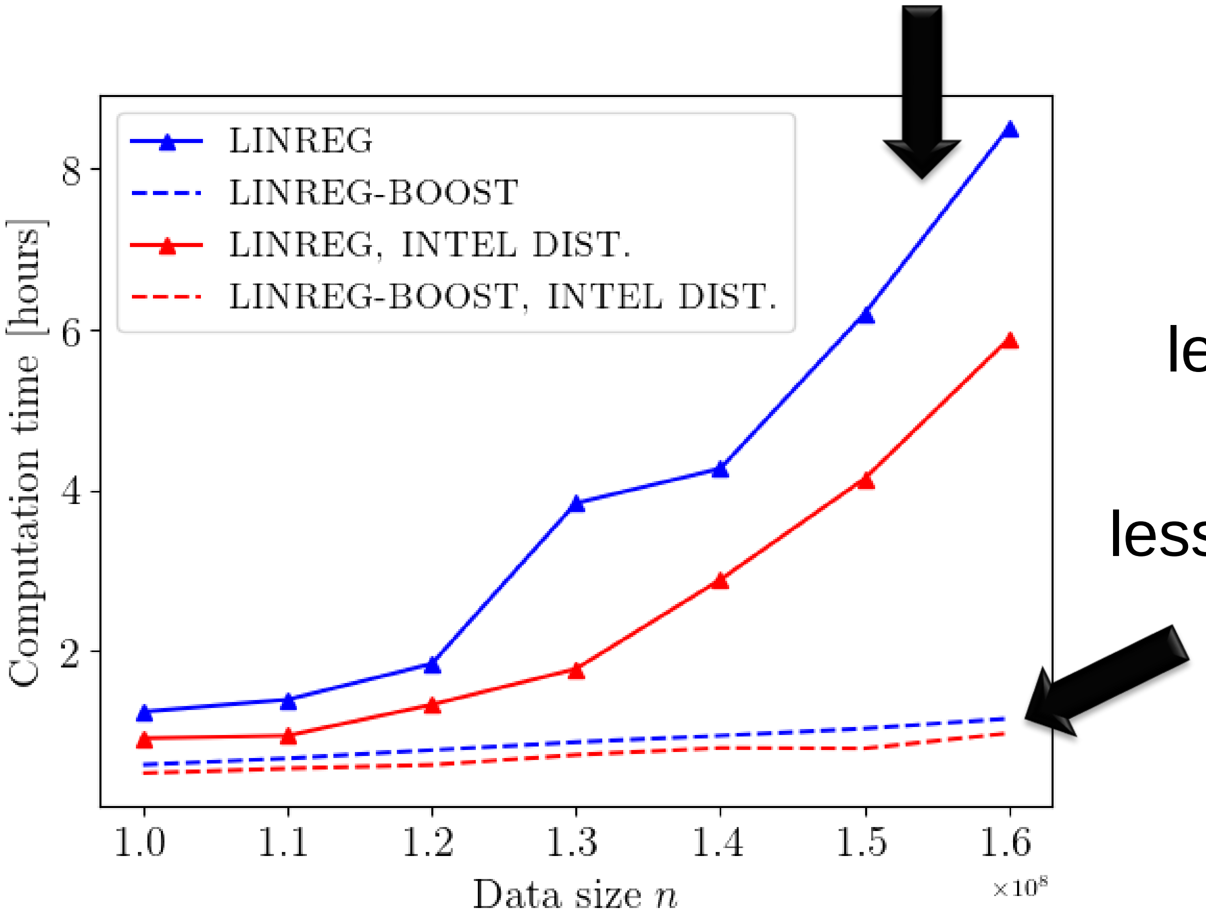
Our compression time + LMS **on small data**

Boosting Computation Time



Improving Space Complexity on Huge Data

Linear regression time is huge due to to memory overload.

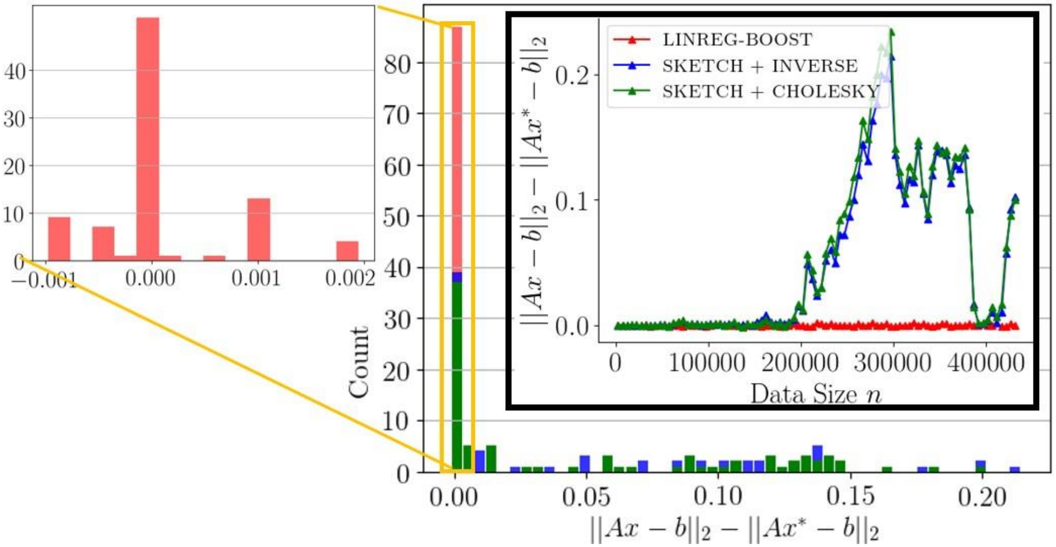


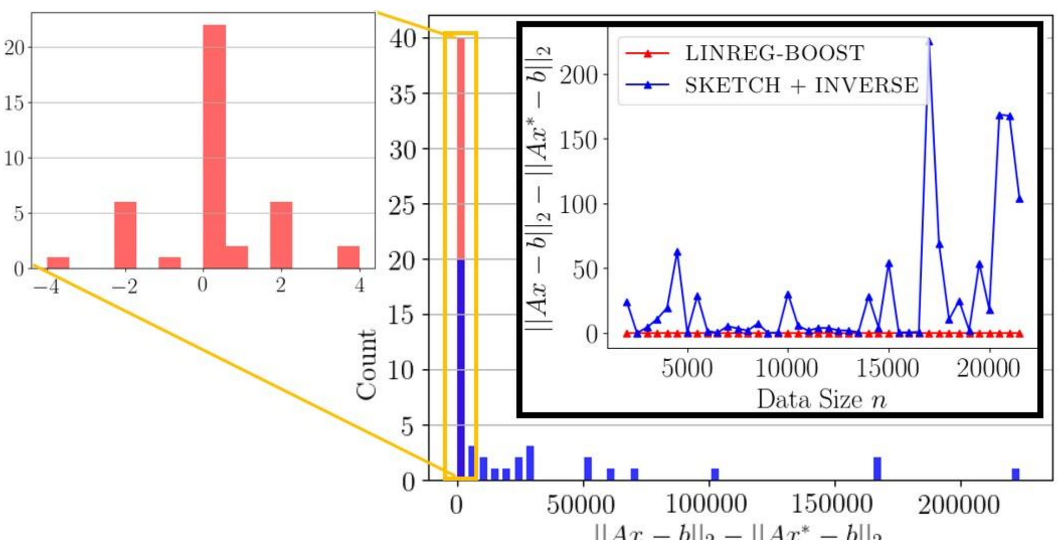
Our smaller data

less memory usage

less computational time

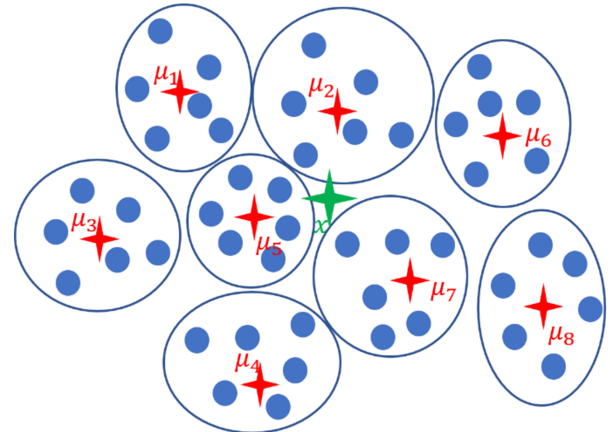
Improving numerical stability vs. other compression schemes





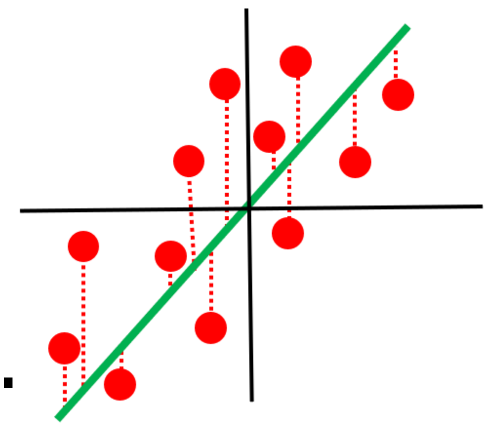
Similar numerical improvement for Ridge / Lasso / Elastic-net

Summary



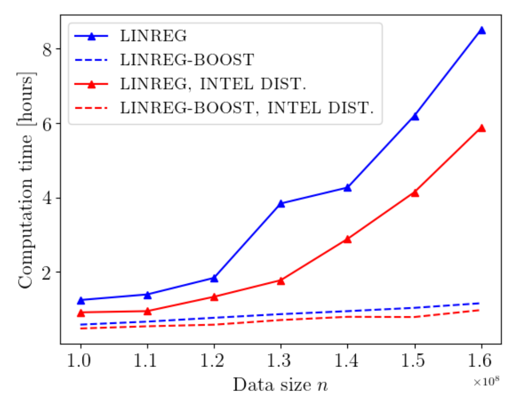
- Computing the *Caratheodory set* in $O(nd)$ time.

- Practically and provably boosting:
  - time complexity of LMS solvers.
  - space complexity of LMS solvers.
  - numerical accuracy of LMS solvers.



- Evaluation on real-world data.

- Full open source code.



Thank you ☺

Open source code