Robust Least Squares / Adversarial training in linear models

Andersen Ang

U.Southampton UK

angms.science

March 24, 2025

1st ver. March 21, 2025

Content

$$\begin{aligned}(\text{RLS}) \ : \quad & \underset{\boldsymbol{x}\in\mathbb{R}^n}{\operatorname{argmin}} \ \max_{\|\boldsymbol{e}_i\|\le\delta} \frac{1}{2}\|(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{x}-\boldsymbol{b}\|_2^2 \\ = \quad & \underset{\boldsymbol{x}\in\mathbb{R}^n}{\operatorname{argmin}} \ \max_{\|\boldsymbol{e}_i\|\le\delta} \sum_{i=1}^{m} \frac{1}{2}\Big(\langle \boldsymbol{a}_i+\boldsymbol{e}_i, \boldsymbol{x}\rangle - b_i\Big)^2\end{aligned}$$

Contents

Ordinary Least Squares

$$(\textbf{OLS}) \ : \ \operatorname*{\mathbf{argmin}}_{\boldsymbol{x}\in\mathbb{R}^n} \ \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2$$

- Consider

$$(\text{OLS}) \ : \ \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \ f(\boldsymbol{x}) \coloneqq \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2$$

  - $\boldsymbol{A} : \mathbb{R}^{m\times n}$ is a given matrix, possible not full rank
  - $\boldsymbol{b} : \mathbb{R}^m$ is a given vector
  - $\boldsymbol{x} \in \mathbb{R}^n$ is the optimization variable
  - The objective function f is a $\mathbb{R}^n \to \mathbb{R}$ mapping

- OLS is
  - the most basic regression model in statistics
  - the most basic linear model in machine learning
  - the most basic convex model in optimization

Ordinary Least Squares

$$(\textbf{OLS}) \ : \ \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2$$

- Summation Form in statistics/machine learning

$$(\text{OLS}) \iff \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \sum_{i=1}^{m} \frac{1}{2}\Big(\langle \boldsymbol{a}_i, \boldsymbol{x}\rangle - b_i\Big)^2 = \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \sum_{i=1}^{m} \frac{1}{2}\ell\Big(\langle \boldsymbol{a}_i, \boldsymbol{x}\rangle - b_i\Big)$$

  - $\boldsymbol{a}_i$ is the ith row of $\boldsymbol{A}$
  - b_i is the ith element (a scalar) of $\boldsymbol{b}$
  - $\langle \boldsymbol{a}_i, \boldsymbol{x}\rangle$ is the inner product between the ith row of $\boldsymbol{A}$ and $\boldsymbol{x}$
  - $\ell = \frac{1}{2}(\cdot)^2$ is called a lost function, and usually $\ell_i(\cdot) \coloneqq \frac{1}{2}\big(\langle \boldsymbol{a}_i, \cdot\rangle - b_i\big)^2$

- Quadratic Programming Form

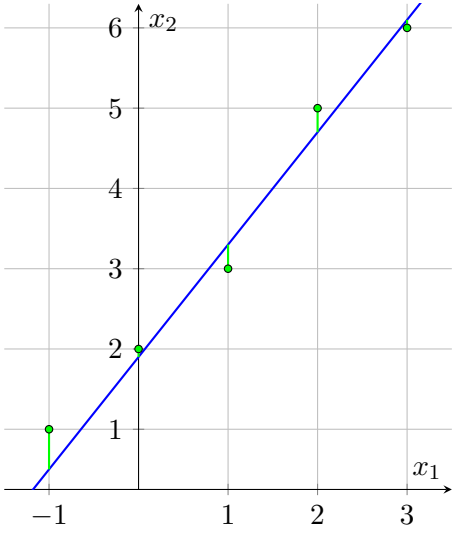
$$(\text{OLS}) \iff \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\langle \boldsymbol{Q}\boldsymbol{x}, \boldsymbol{x}\rangle - \langle \boldsymbol{p}, \boldsymbol{x}\rangle + \frac{1}{2}\|\boldsymbol{b}\|_2^2$$

  - $\boldsymbol{Q} = \boldsymbol{A}^\top \boldsymbol{A}$
  - $\boldsymbol{p} = \boldsymbol{A}^\top \boldsymbol{b}$

here are 90 pages more details

Contents

Picture of OLS

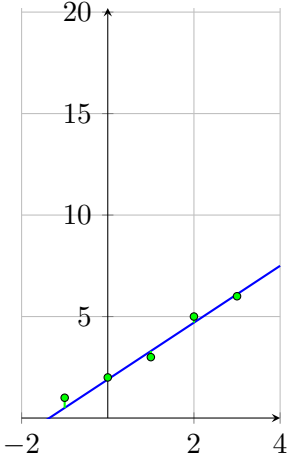


$$\text{(OLS)} \iff \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \sum_{i=1}^{m} \frac{1}{2}\Big(\langle \boldsymbol{a}_i, \boldsymbol{x}\rangle - b_i\Big)^2$$

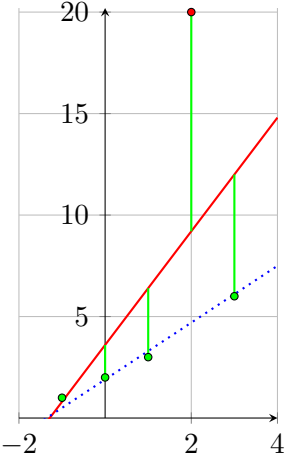
- We are giving a bunch of points (2D points here)
- Solving OLS is to find the blue line
- The blue line is the best fit to these points
- The word "best" refers to the sum of the length of the green segments is the shortest
- Geometrically,

  solve OLS $\iff$ move the line for the shortest sum

Motivation of Robust LS: OLS is sensitive to outlier



- Data is “ok”
- The line fits all points well “on average”



- A point is an outlier
- The line is dragged by that outlier
- To fit the line we should “ignore” the outlier

Contents

Robust Least Squares

$$(\textbf{OLS}) \ : \ \operatorname*{\mathbf{argmin}}_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2$$

$$(\text{RLS}) \ : \ \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \ \max_{\|\boldsymbol{e}_i\|\le\delta} \frac{1}{2}\|(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{x}-\boldsymbol{b}\|_2^2$$

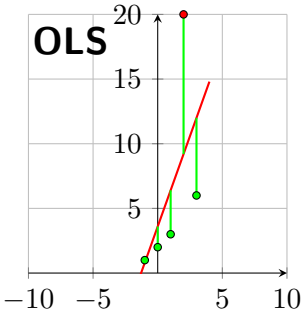
- What's new: $\max$ and $\boldsymbol{E}$
  - $\boldsymbol{E}:\mathbb{R}^{m\times n}$ is an unknown matrix, also an optimization variable here
  - $\boldsymbol{e}_i:\mathbb{R}^n$ is the ith row of $\boldsymbol{E}$
  - $\delta\in\mathbb{R}$ is the max perturbation radius, it tells the max magnitude of the perturbation

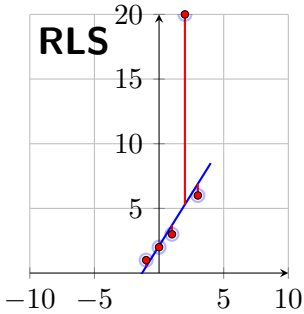
- May be easier to understand RLS in summation form

$$(\text{RLS}) \iff \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \ \max_{\|\boldsymbol{e}_i\|\le\delta} \sum_{i=1}^{m} \frac{1}{2}\Big(\langle \boldsymbol{a}_i+\boldsymbol{e}_i,\boldsymbol{x}\rangle - b_i\Big)^2$$

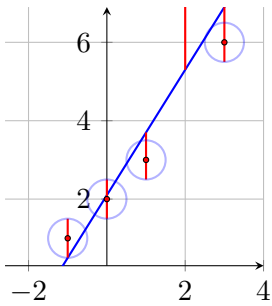
  - think of $\boldsymbol{a}_i$ is "true value" and $\boldsymbol{e}_i$ is "additive noise"
  - think of $\hat{\boldsymbol{a}}_i=\boldsymbol{a}_i+\boldsymbol{e}_i$ is $\boldsymbol{a}_i$ corrupted by noise
  - $\|\boldsymbol{e}_i\|\le\delta$ means how large is the noise's size
  - we don't know the exact value of noise, so we consider all the possible $\boldsymbol{e}_i$
  - $\max_{\|\boldsymbol{e}_i\|\le\delta}$ means we take the worst case among all the possible noise

OLS vs RLS





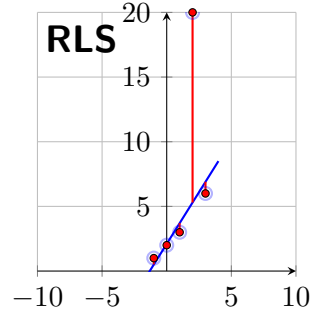
- Data points have "zero volume" (I draw them thicker for visibility)
- The vertical line touches the points

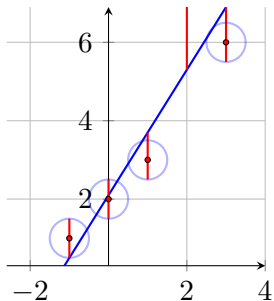


In RLS

- Data points are "points with volume"
- Circle radius = perturbation radius δ
- We are now considering the largest sum of red length touching circles
- "largest sum" contains the meaning of the "worst case"
- using worst case formulation, we magically ignore the outlier !

Why RLS ignores the outlier?





- By definition: outlier are far away $\implies$ the length is large
  - If δ is small, the perturbation $\|e_i\| \leq \delta$ have limited impact on the length
- For inliners (non-outliers), the length is smaller
  - the perturbation $\|e_i\| \leq \delta$ have larger impact on the length
- If the line is moving away from the inliners, the inliners error

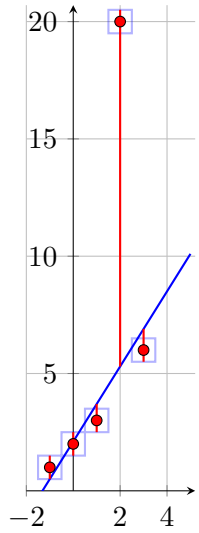
$$\text{inliners error} = \max_{\|e_i\| \leq \delta} \sum_{i \in \text{inliners}} \frac{1}{2}\Big(\langle a_i + e_i, x\rangle - b_i\Big)^2$$

will go up much faster than the outliners error

$$\text{outliners error} = \max_{\|e_i\| \leq \delta} \sum_{i \notin \text{inliners}} \frac{1}{2}\Big(\langle a_i + e_i, x\rangle - b_i\Big)^2 \approx \sum_{i \notin \text{inliners}} \frac{1}{2}\Big(\langle a_i, x\rangle - b_i\Big)^2$$

therefore the $\operatorname*{argmin}_x$ will pull the line towards inliners

ℓ_∞-norm version of RLS



$$\text{(RLS)} \;:\; \underset{\boldsymbol{x}\in\mathbb{R}^n}{\operatorname{argmin}} \max_{\|\boldsymbol{e}_i\|\le\delta} \frac{1}{2}\|(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{x}-\boldsymbol{b}\|_2^2$$

- Previously we used $\|\boldsymbol{e}_i\|_2 \le \delta$

$$\|\boldsymbol{v}\|_2 = \sqrt{v_1^2 + v_2^2 + \dots}$$

- Here we used $\|\boldsymbol{e}_i\|_\infty \le \delta$

$$\|\boldsymbol{v}\|_\infty = \max_i \Big\{|v_1|, |v_2|, \dots\Big\}$$

- There is also $\|\boldsymbol{e}_i\|_1 \le \delta$

$$\|\boldsymbol{v}\|_1 = |v_1| + |v_2| + \dots$$

Contents

Min-max Optimization

$$\text{(RLS)} \;:\; \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \max_{\|\boldsymbol{e}_i\|\le\delta} \frac{1}{2}\|(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{x}-\boldsymbol{b}\|_2^2 \;=\; \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \max_{\|\boldsymbol{e}_i\|\le\delta} \sum_{i=1}^{m} \frac{1}{2}\Big(\langle \boldsymbol{a}_i+\boldsymbol{e}_i, \boldsymbol{x}\rangle - b_i\Big)^2$$

- RLS is an example of min-max optimization

$$\text{(Min-max)} \;:\; \min_{\boldsymbol{x}} \max_{\boldsymbol{y}} f(\boldsymbol{x},\boldsymbol{y})$$

- Goal of solving min-max problem: find saddle point $(\boldsymbol{x}^*, \boldsymbol{y}^*)$

$$f(\boldsymbol{x}^*, \boldsymbol{y}) \;\le\; f(\boldsymbol{x}^*, \boldsymbol{y}^*) \;\le\; f(\boldsymbol{x}, \boldsymbol{y}^*)$$

- Where will you see Min-max optimization
  - Robust optimization
  - Dual-based optimization
  - Game Theory
  - Machine Learning using robustness and adversarial training

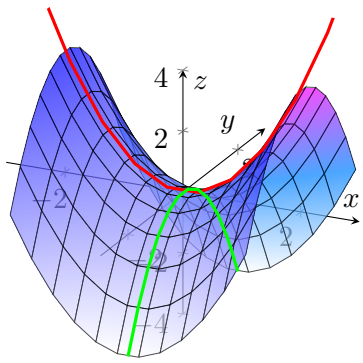
What is a saddle point

$$\text{(Min-max)} \;:\; \min_{\boldsymbol{x}} \max_{\boldsymbol{y}} f(\boldsymbol{x}, \boldsymbol{y})$$

- Goal of solving min-max problem: find saddle point $(\boldsymbol{x}^*, \boldsymbol{y}^*)$

$$f(\boldsymbol{x}^*, \boldsymbol{y}) \;\leq\; f(\boldsymbol{x}^*, \boldsymbol{y}^*) \;\leq\; f(\boldsymbol{x}, \boldsymbol{y}^*)$$

- $\boldsymbol{x}^*$ is minimizer if you view along x-axis
- $\boldsymbol{y}^*$ is maximizer if you view along y-axis
- $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ is x-min y-max if we view x-axis y-axis together



General theory of Min-max optimization

(Min-max) : $\min_{\boldsymbol{x}\in\mathcal{X}} \max_{\boldsymbol{y}\in\mathcal{Y}} f(\boldsymbol{x},\boldsymbol{y})$ Saddle point: $f(\boldsymbol{x}^*,\boldsymbol{y}) \;\le\; f(\boldsymbol{x}^*,\boldsymbol{y}^*) \;\le\; f(\boldsymbol{x},\boldsymbol{y}^*)$

f is $\mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$

- **Von Neumann's Minimax Theorem** If f is convex in $\boldsymbol{x}$ and concave in $\boldsymbol{y}$, then

$$\min_{\boldsymbol{x}\in\mathcal{X}} \max_{\boldsymbol{y}\in\mathcal{Y}} f(\boldsymbol{x},\boldsymbol{y}) \;=\; \max_{\boldsymbol{y}\in\mathcal{Y}} \min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x},\boldsymbol{y})$$

- **Subgradient 1st-order optimality**

$$\mathbf{0}_n \in \partial_{\boldsymbol{x}}\Big(f(\boldsymbol{x}^*,\boldsymbol{y}^*) + \iota_{\mathcal{X}}(\boldsymbol{x}^*,\boldsymbol{y}^*)\Big), \qquad \mathbf{0}_m \in \partial_{\boldsymbol{y}}\Big(f(\boldsymbol{x}^*,\boldsymbol{y}^*) + \iota_{\mathcal{Y}}(\boldsymbol{x}^*,\boldsymbol{y}^*)\Big)$$

- **Alternating Optimization: Gradient Descent-Ascent (GDA)**

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \alpha\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_k,\boldsymbol{y}_k) \;\text{ followed by } \mathrm{proj},\ \mathrm{prox} \text{ ect}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + \alpha\nabla_{\boldsymbol{y}} f(\boldsymbol{x}_k,\boldsymbol{y}_k) \;\text{ followed by } \mathrm{proj},\ \mathrm{prox} \text{ ect}$$

Contents

How to solve RLS

$$\text{(RLS)} \;:\; \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \max_{\|\boldsymbol{e}_i\|\le\delta} \frac{1}{2}\|(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{x}-\boldsymbol{b}\|_2^2 \;=\; \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \max_{\|\boldsymbol{e}_i\|\le\delta} \sum_{i=1}^{m} \frac{1}{2}\Big(\langle \boldsymbol{a}_i+\boldsymbol{e}_i,\boldsymbol{x}\rangle - b_i\Big)^2$$

- Alternating Optimization: repeat

1. Update $\boldsymbol{x}$ with a fix $\boldsymbol{E}$
2. Update $\boldsymbol{E}$ with a fix $\boldsymbol{x}$

- x-update: gradient descent (here are 90 pages more details)

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \alpha\Big((\boldsymbol{A}+\boldsymbol{E})^\top(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{x}_k - (\boldsymbol{A}+\boldsymbol{E})^\top\boldsymbol{b}\Big)$$

- E-update: projected gradient ascent

$$\boldsymbol{e}_{i,k+1} = \operatorname{proj}_{\|\boldsymbol{e}\|\le\delta}\Big(\boldsymbol{e}_{i,k} + \alpha\nabla_{\boldsymbol{e}}\frac{1}{2}\Big(\langle \boldsymbol{a}_i+\boldsymbol{e},\boldsymbol{x}\rangle - b_i\Big)^2\Big)$$

- projected gradient DEscent (change DEscent to AScent)
- proj onto ℓ_2-ball
- proj onto ℓ_1-ball

Find $\nabla_{\boldsymbol{x}} \left((\boldsymbol{a} + \boldsymbol{x})^\top \boldsymbol{c} - b \right)^2$

$$f(\boldsymbol{x}) = \left((\boldsymbol{a} + \boldsymbol{x})^\top \boldsymbol{c} - b \right)^2 = \left(\boldsymbol{a}^\top \boldsymbol{c} + \boldsymbol{x}^\top \boldsymbol{c} - b \right)^2$$

- Let $u = \boldsymbol{a}^\top \boldsymbol{c} + \boldsymbol{x}^\top \boldsymbol{c} - b$, then $f(\boldsymbol{x}) = u^2$
- Chain rule

$$\frac{\partial f}{\partial \boldsymbol{x}} = 2u \frac{\partial u}{\partial \boldsymbol{x}}$$

- With $\dfrac{\partial u}{\partial \boldsymbol{x}} = \boldsymbol{c}$

$$\frac{\partial f}{\partial \boldsymbol{x}} = 2u\boldsymbol{c} = 2 \left(\boldsymbol{a}^\top \boldsymbol{c} + \boldsymbol{x}^\top \boldsymbol{c} - b \right) \boldsymbol{c} = 2 \left((\boldsymbol{a} + \boldsymbol{x})^\top \boldsymbol{c} - b \right) \boldsymbol{c}$$

- Final gradient:

$$\nabla_{\boldsymbol{x}} \frac{1}{2} \left((\boldsymbol{a} + \boldsymbol{x})^\top \boldsymbol{c} - b \right)^2 = \left((\boldsymbol{a} + \boldsymbol{x})^\top \boldsymbol{c} - b \right) \boldsymbol{c} = 2f(\boldsymbol{x})\boldsymbol{c}$$

Alternating optimization solve RLS

$$\text{(RLS)} \;:\; \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \max_{\|\boldsymbol{e}_i\|\le\delta} \frac{1}{2}\|(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{x}-\boldsymbol{b}\|_2^2 \;=\; \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} \max_{\|\boldsymbol{e}_i\|\le\delta} \sum_{i=1}^{m} \frac{1}{2}\Big(\langle \boldsymbol{a}_i+\boldsymbol{e}_i, \boldsymbol{x}\rangle - b_i\Big)^2$$

- x-update:

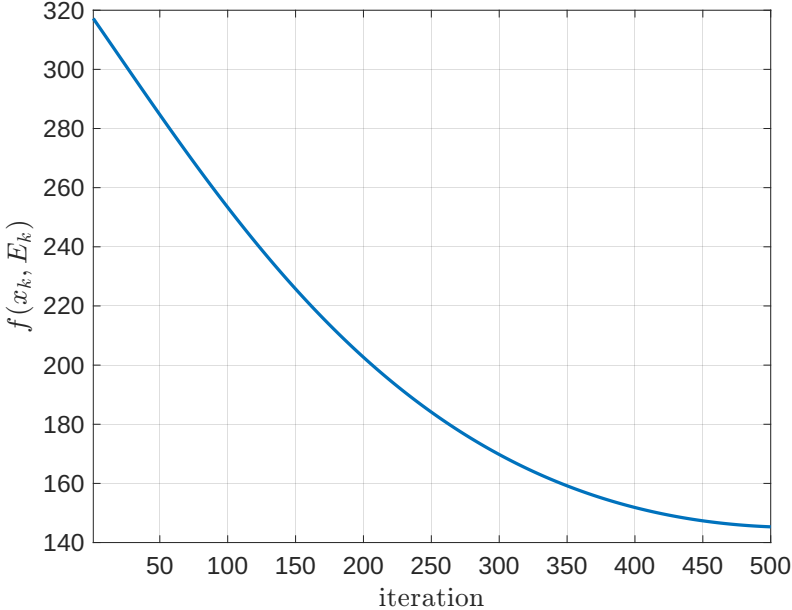
$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \alpha\Big((\boldsymbol{A}+\boldsymbol{E})^\top(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{x}_k - (\boldsymbol{A}+\boldsymbol{E})^\top\boldsymbol{b}\Big)$$

- E-update:

$$\hat{\boldsymbol{e}}_{i,k+1} = \boldsymbol{e}_{i,k} + \alpha\Big(\langle \boldsymbol{a}_i+\boldsymbol{e}_{i,k}, \boldsymbol{x}_k\rangle - b_i\Big)\boldsymbol{x}_k$$

$$\boldsymbol{e}_{i,k+1} = \operatorname{proj}_{\|\boldsymbol{e}\|\le\delta}\Big(\hat{\boldsymbol{e}}_{i,k+1}\Big) = \frac{\hat{\boldsymbol{e}}_{i,k+1}}{\max\Big\{1, \dfrac{\|\hat{\boldsymbol{e}}_{i,k+1}\|}{\delta}\Big\}}$$

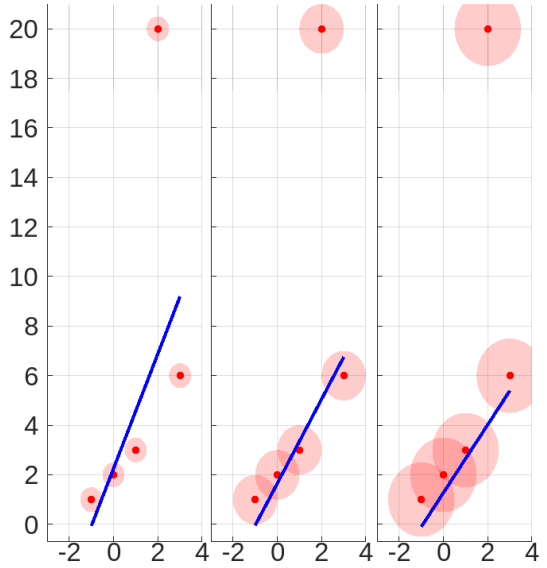
Quick-dirty-done MATLAB code



Effect of the radius δ

$$(\text{RLS}) \ : \ \underset{\boldsymbol{x}\in\mathbb{R}^n}{\operatorname{argmin}} \ \max_{\|\boldsymbol{e}_i\|\le\delta} \frac{1}{2}\|(\boldsymbol{A}+\boldsymbol{E})\boldsymbol{x}-\boldsymbol{b}\|_2^2$$

- If $\delta \to 0$, then $\boldsymbol{E} \to \boldsymbol{0}$, and RLS $\to$ OLS
- Decreasing δ makes RLS back to OLS
- Increasing δ makes the inliner "more important"



Last page - summary

Ordinary Least Squares

Motivation of Robust Least Sqaures

Robust Least Squares

Min-max optimization

Alternating optimization

End of document