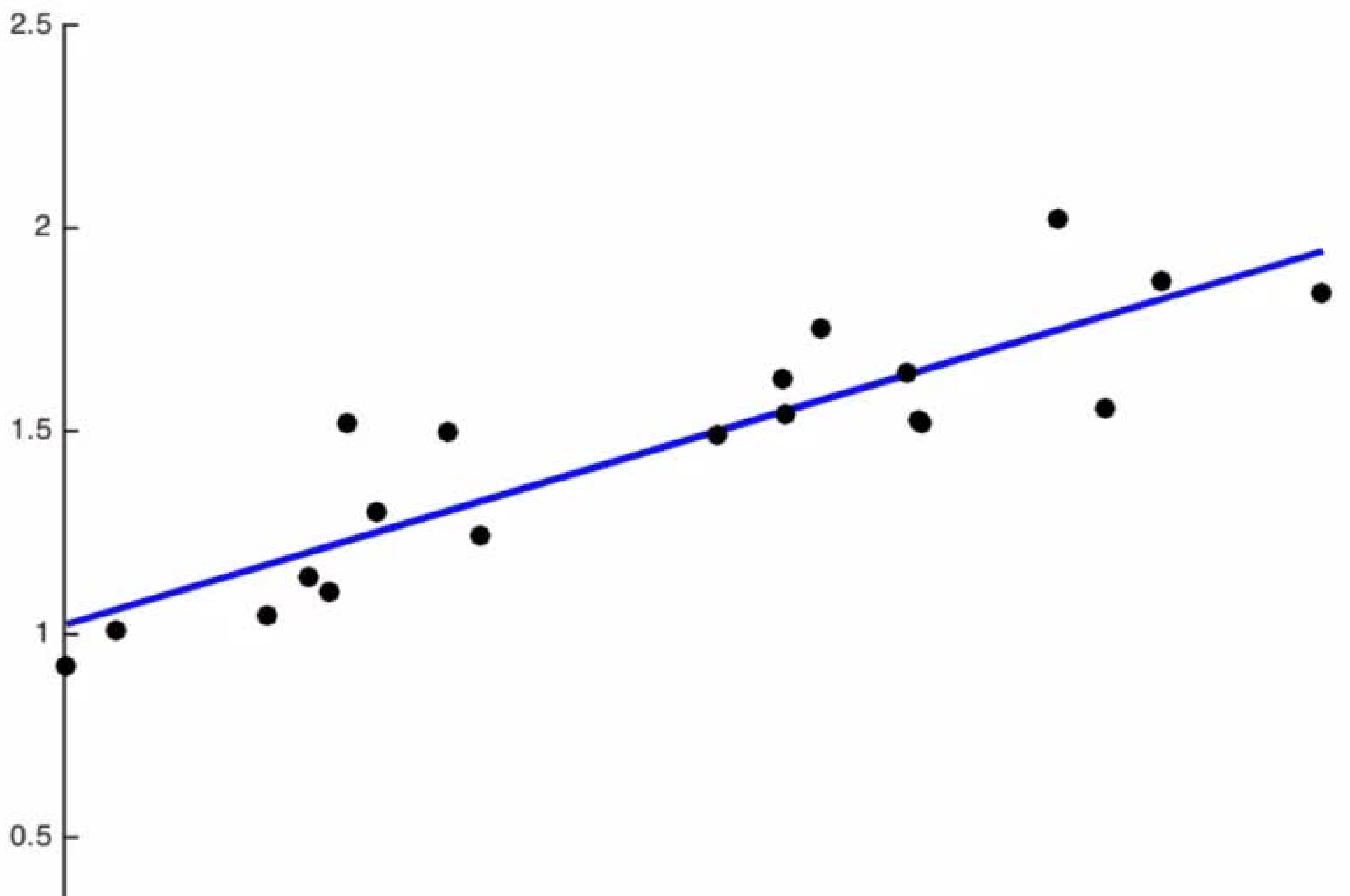
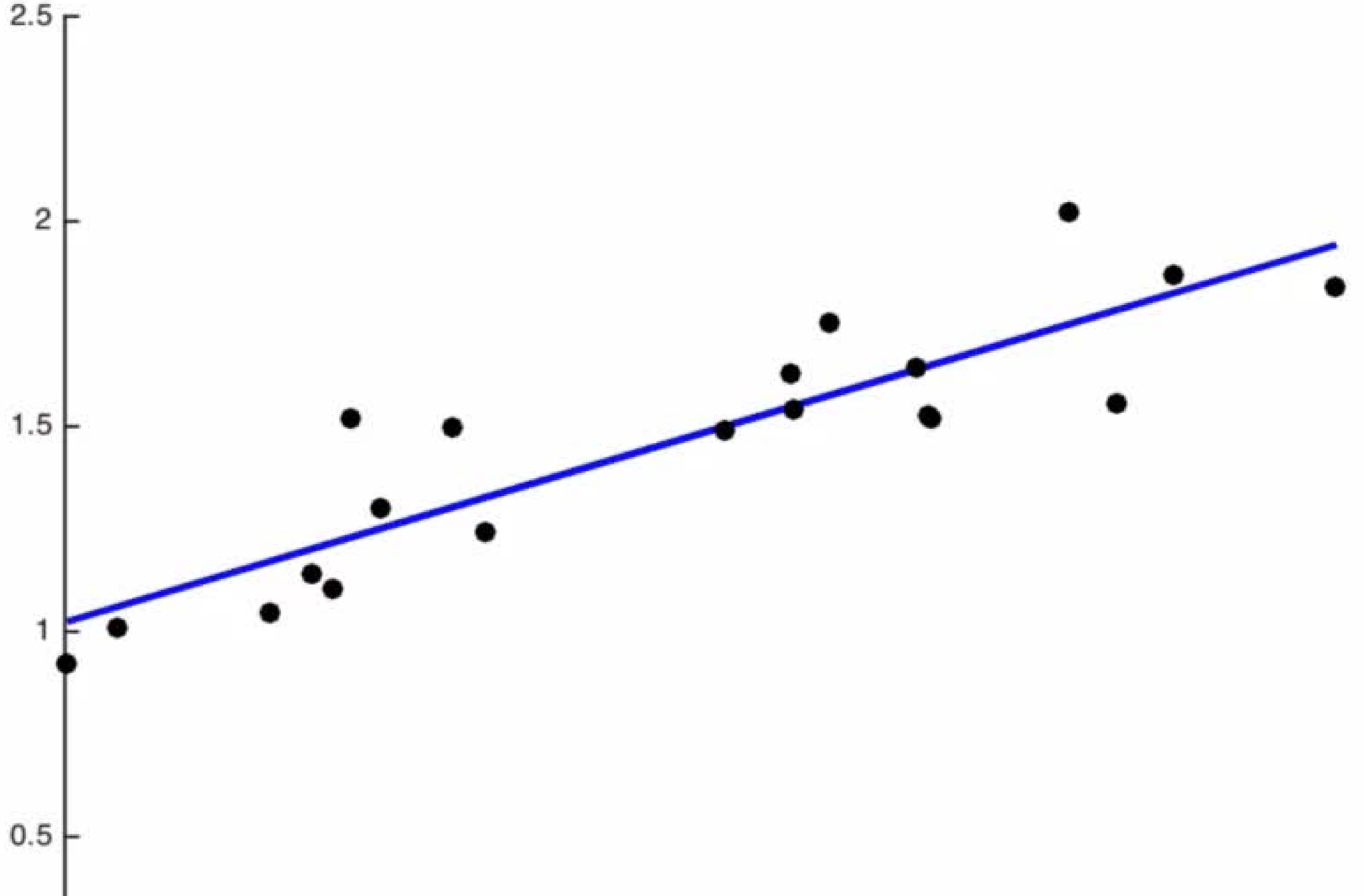
Trimming Linear and Nonlinear Models.

Aleksandr Aravkin[1], Damek Davis[2], and Peng Zheng[3]

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \ \frac{1}{n} \sum_{i=1}^{n} \|a_i^T x - b_i\|^2$$

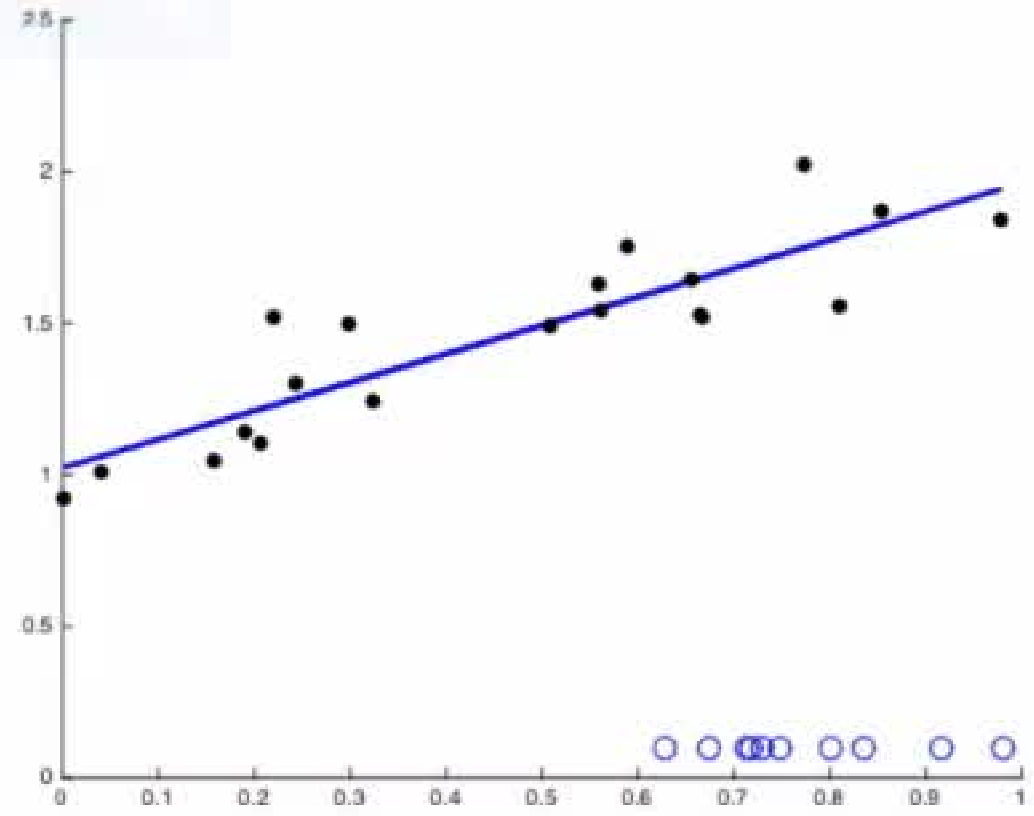


$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \ \frac{1}{n} \sum_{i=1}^{n} \|a_i^T x - b_i\|^2$$



[2]Cornell University, Operations Research and Information Engineering
[3]University of Washington, Applied Math

$$\underset{x\in\mathbb{R}^d}{\text{minimize}}\ \frac{1}{n}\sum_{i=1}^{n}\|a_i^T x - b_i\|^2$$



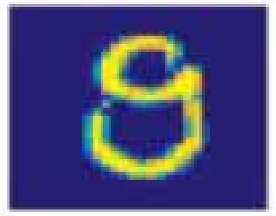
Application: Trimmed Digit Classification

$$\underset{X \in \mathbb{R}^{n \times K}}{\text{minimize}} \ \frac{1}{n}\sum_{i=1}^{n} w_i \left(\text{LSE}(Xa_i) - a_i^T X y_i\right) + \frac{\lambda}{2n}\|X\|_F^2 + r_1(w).$$

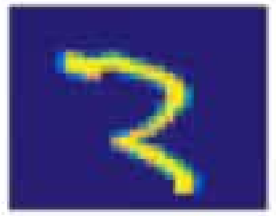
- K-class classification with multinomial logistic regression
- Each column of X is a set of weights for recognizing the Kth class.
- If example i is in class 1, then $a_i^T X y_i = x_1^T a_i$, so want

$$\text{LSE}(Xa_i) = \log\left(\sum_{s=1}^{K} \exp(x_s^T a_i)\right) \approx x_1^T a_i.$$

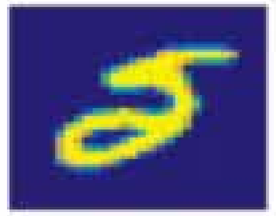
- Formulation ties to make $x_1^T a_i >> x_k^T a_i$ for $k \neq 1$.
- **Dataset:** MNIST's $n = 60000$ handwritten digits (0-9)
- Trim with $h = 0.99n$, find 1% worst outliers. Labels?



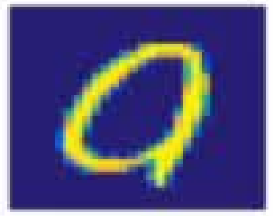
?



?

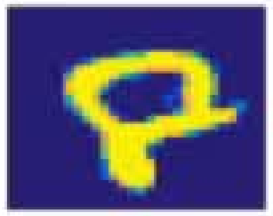


?



?

?

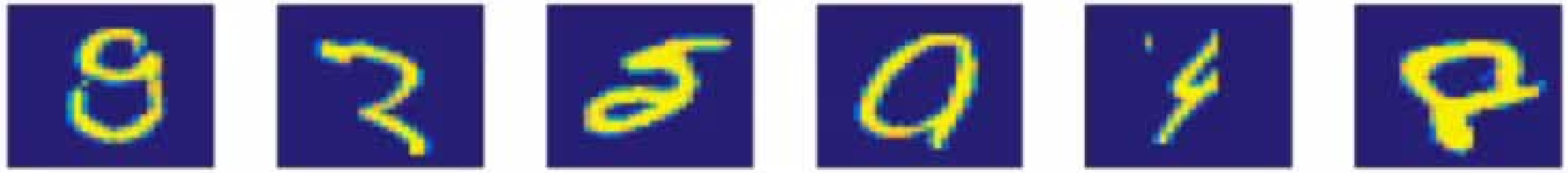


?

Application: Trimmed Digit Classification

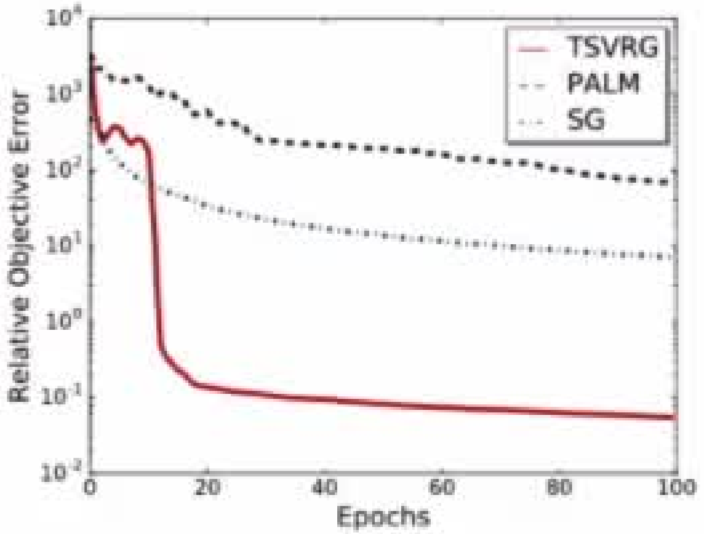
$$\underset{X \in \mathbb{R}^{n \times K}}{\text{minimize}} \; \frac{1}{n} \sum_{i=1}^{n} w_i \left(\text{LSE}(X v_i) - v_i^T X y_i \right) + \frac{\lambda}{2n} \|X\|_F^2 + r_1(w).$$

- K-class classification with multinomial logistic regression
- **Dataset:** MNIST's $n = 60000$ handwritten digits (0-9)
- Trim with $h = 0.99n$, find 1% worst outliers. Labels?

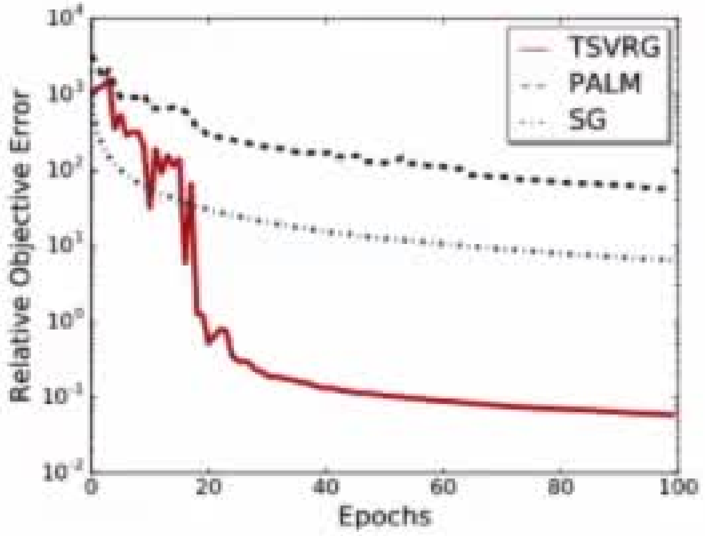


9 3 5 9 4 8

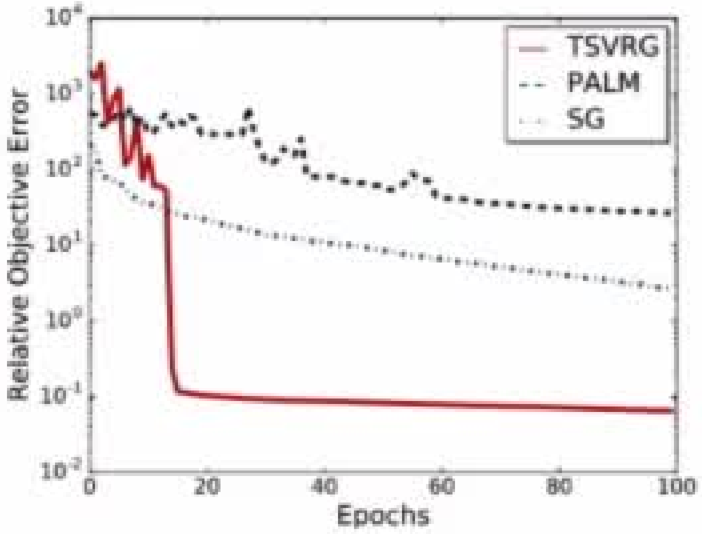
Trimmed MNIST with PALM, SGD, and TSVRG



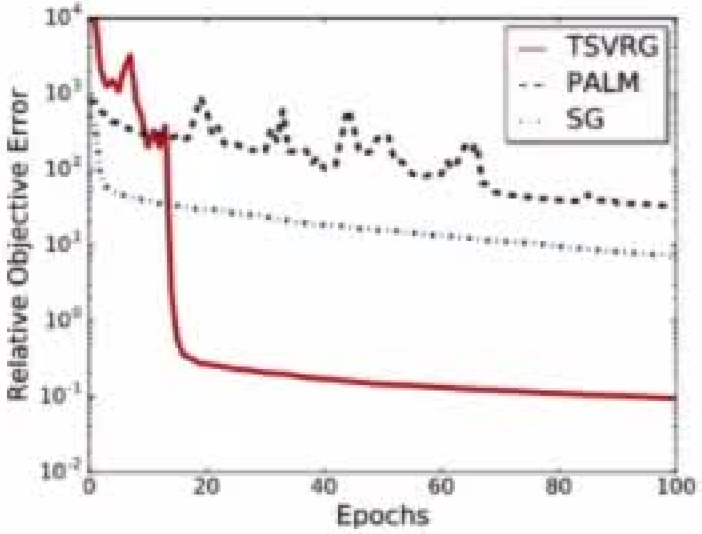
(a) 10% contamination



(b) 20% contamination



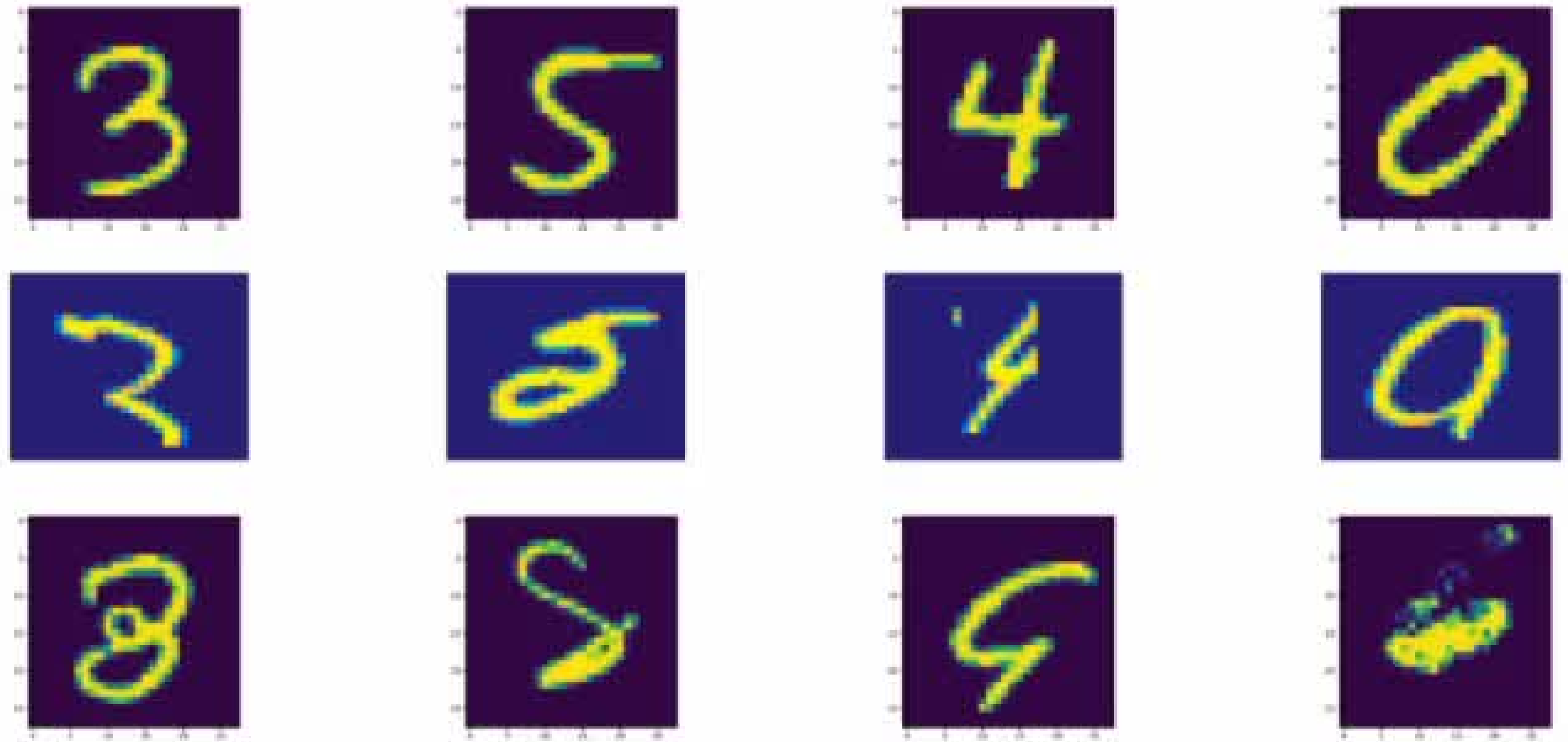
(c) 30% contamination



(d) 40% contamination

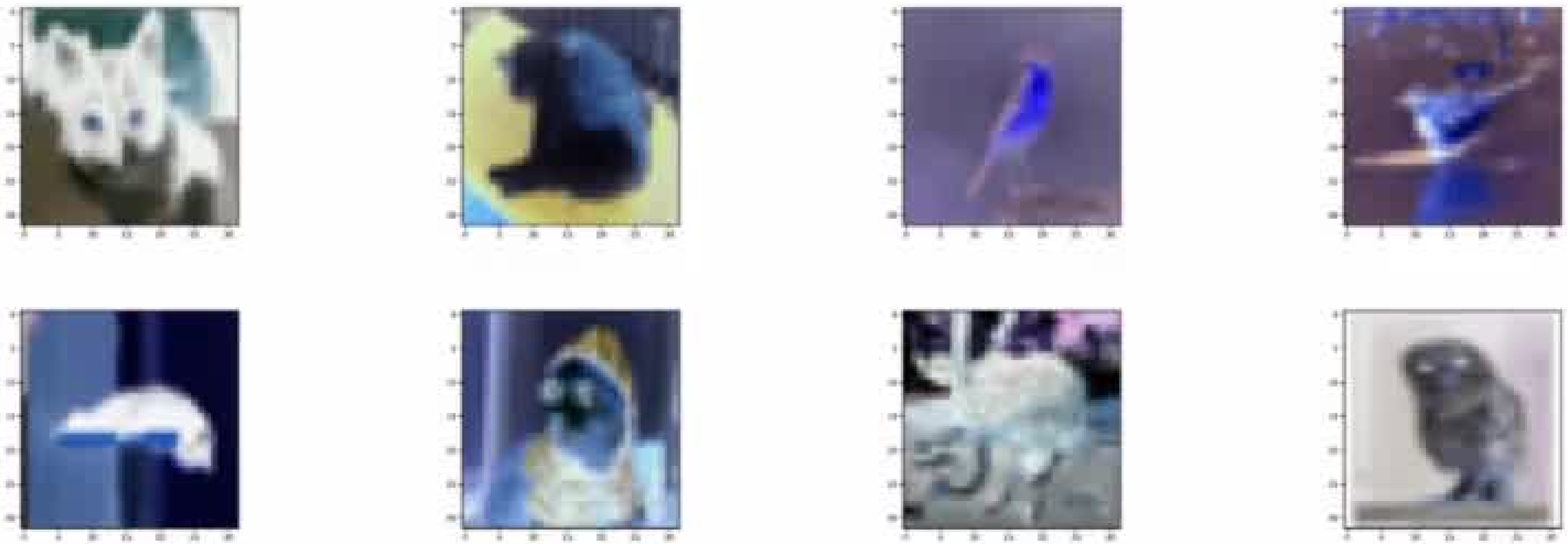
Trimming Neural Nets

- Peng Zheng implemented TSVRG in TensorFlow for several types of nets.
- Neural nets are more expressive, so the outliers are more extreme.



Trimming Convolutional Neural Nets

- Convolutional neural nets (CNNs) are used to classify images.
- CNNs control model complexity by connecting output neurons to small regions of input space. They add layers that 'downsample' inputs.
- Separating cats and birds: inliers vs. outliers



Application: Trimmed Homography Estimation

(a) Image 1

(b) Image 2

- **Task:** Estimate *homography* $H \in \mathbb{R}^{3\times 3}$ between two images of the same scene
- Given points (u_1, v_1) in image 1 and (u_2, v_2) in image 2, we have

$$H \begin{bmatrix} u_1 & v_1 & 1 \end{bmatrix}^T = \begin{bmatrix} u_2 & v_2 & 1 \end{bmatrix}^T$$

- Given a set of correspondences B_1 and B_2, get H via

$$\min_{\|H\|_F = 1} \|HB_1 - B_2\|_F^2.$$

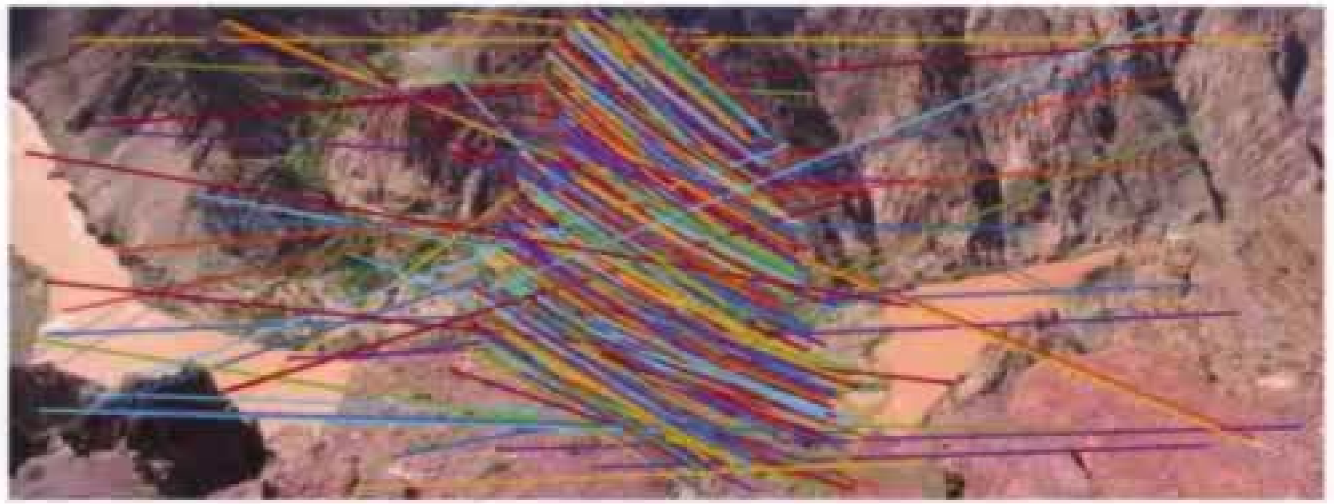
- Nonconvex problem!

Application: Trimmed Homography Estimation

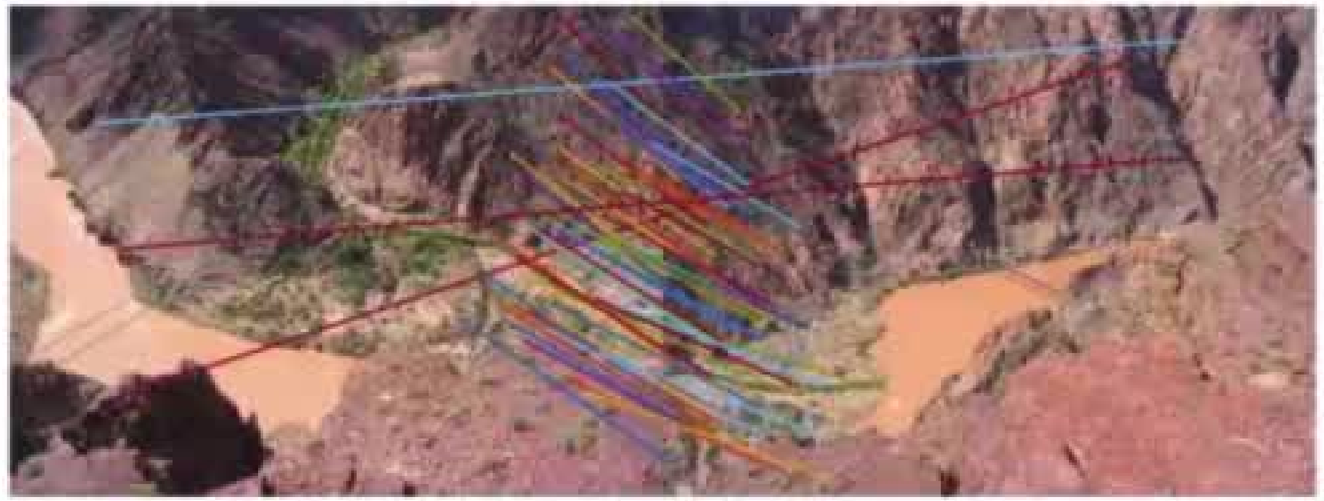
- Only need four correspondences to get a homography.
- Problem: one bad correspondence can completely ruin homography
- Trim instead:

$$\underset{w \in \Delta^k, \|H\|_F = 1}{\text{minimize}} \sum_{i=1}^{n} w_i \|Hb_{1,i} - b_{2,i}\|^2$$

Application: Trimmed Homography Estimation



(a) Tentative SIFT matches



(b) Best 10% matches found by TSVRG (We choose $h = .1n$)

- Matches extremely noisy
- In practice, RANSAC (brute force) used to find four inlying points

Application: Trimmed Homography Estimation

- Only need four correspondences to get a homography.
- Problem: one bad correspondence can completely ruin homography
- Trim instead:

$$\underset{w \in \Delta^h, \|H\|_F = 1}{\text{minimize}} \sum_{i=1}^{n} w_i \|Hb_{1,i} - b_{2,i}\|^2$$