Robust AI at Microsoft Research

Jerry Li (MSR AI)

Talk organization

- Part 1: Robustness at training time
  - What happens when the training set has outliers?

- Part 2: Robustness at test time
  - What happens when your adversary tries to fool your model?

Robustness at Train Time

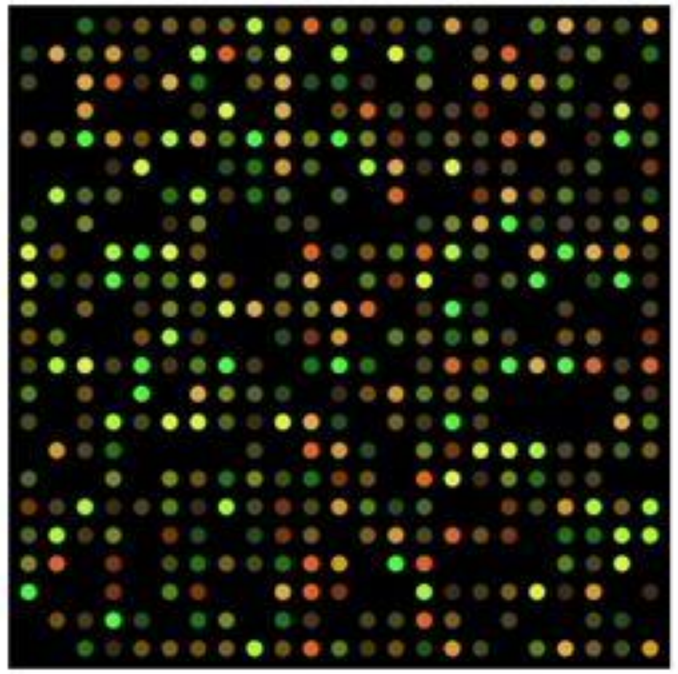
Two motivating examples

Two motivating examples

Genetic data

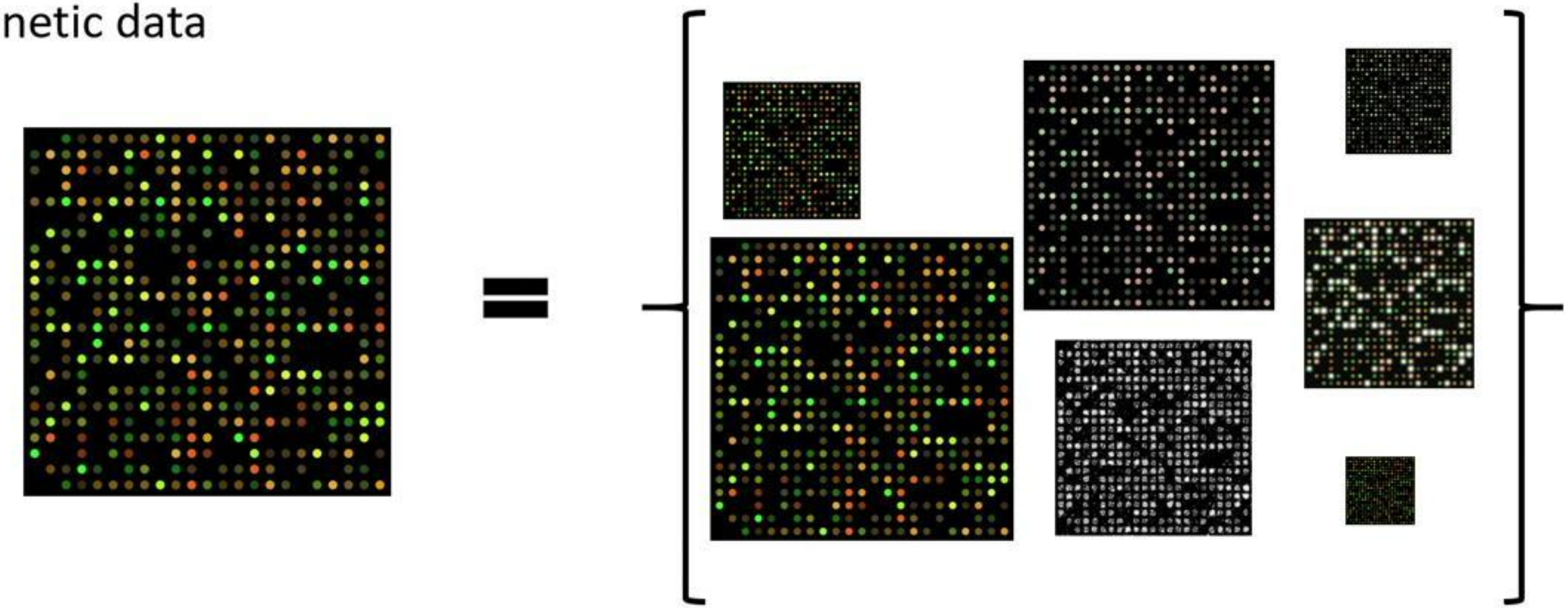
Two motivating examples

Genetic data



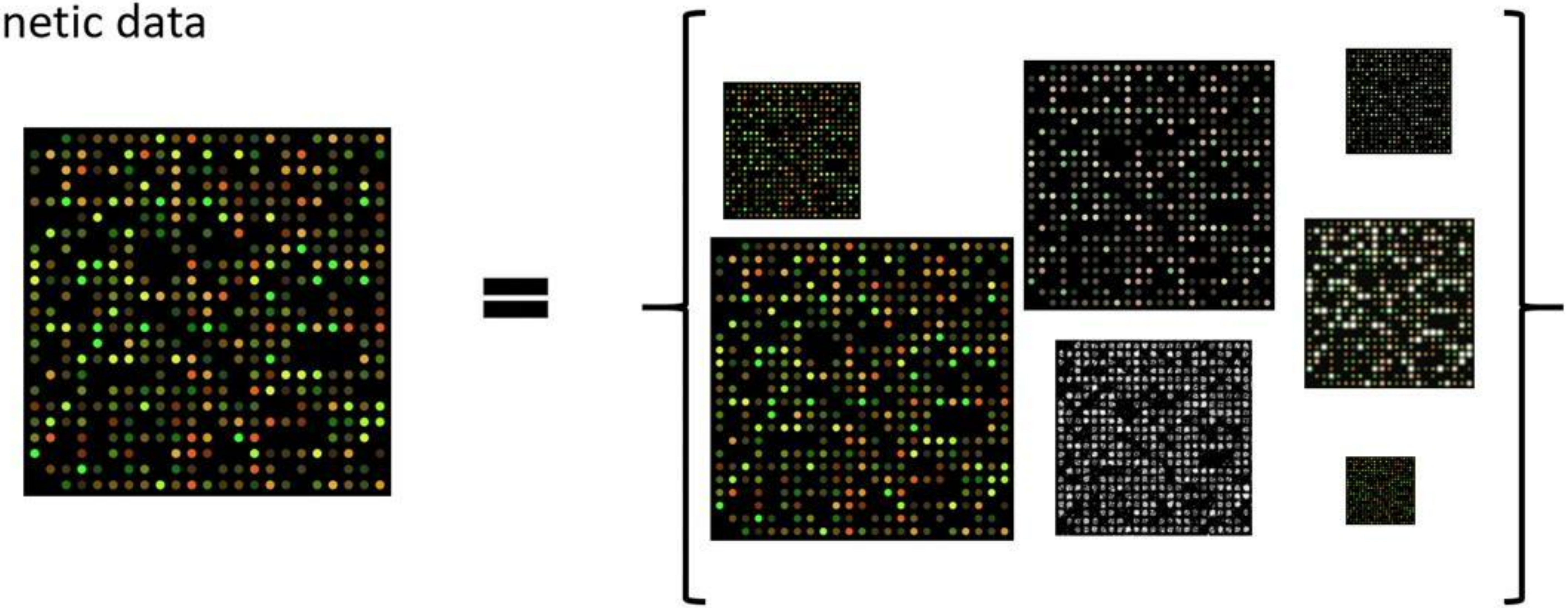
Two motivating examples

Genetic data



Two motivating examples

Genetic data



Data is often heterogeneous, causing uncontrolled systematic noise

Two motivating examples

Two motivating examples

Data poisoning / Adversarial machine learning

Figure from [Gu, Dolan-Gavitt, Garg '17]

Two motivating examples

Data poisoning / Adversarial machine learning

Figure from [Gu, Dolan-Gavitt, Garg '17]

Data can come from untrusted / tampered sources

Two motivating examples

Two motivating examples

Large data sets are often inherently noisy

Two motivating examples

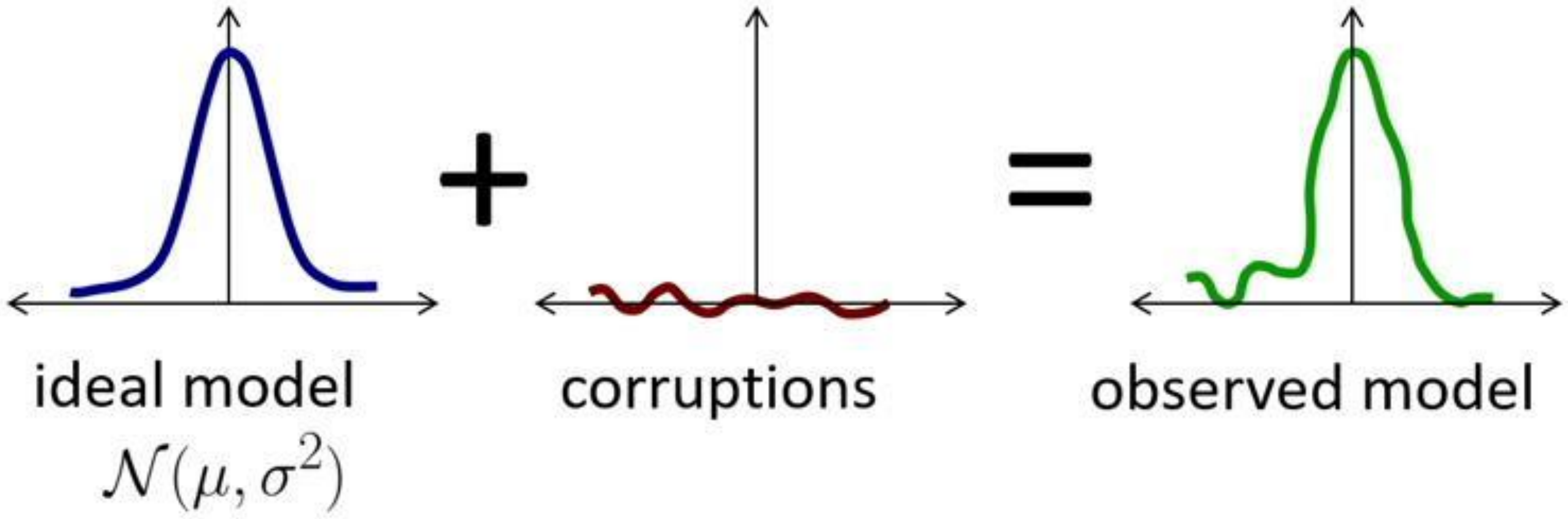
Large data sets are often inherently noisy

How can we learn from noisy high dimensional data?

Two motivating examples

Large data sets are often inherently noisy

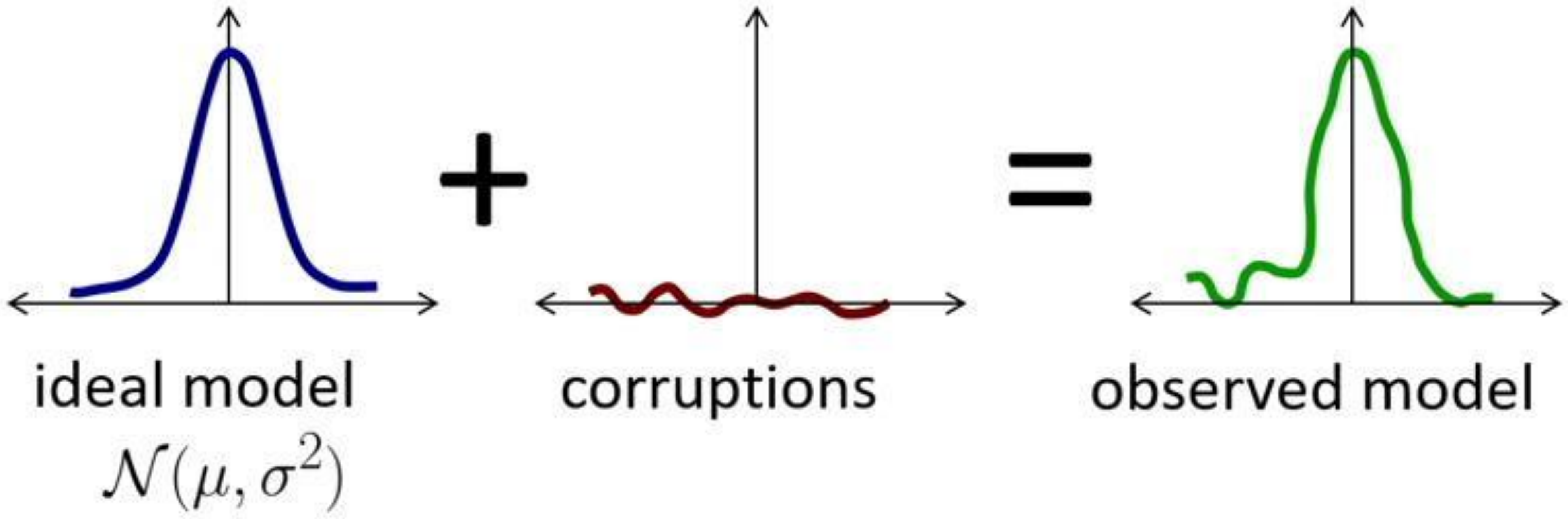
How can we learn from noisy high dimensional data?



Two motivating examples

Large data sets are often inherently noisy

How can we learn from noisy high dimensional data?



Challenge: Develop algorithms which are provably robust to worst case noise

Robust statistics

[Huber], [Tukey] ’60s

Robust statistics

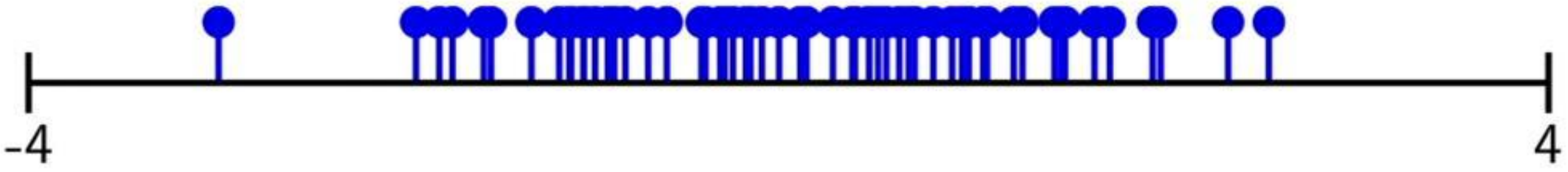
[Huber], [Tukey] '60s

- Given samples from a distribution, where an adversary has moved an ε-fraction of the points arbitrarily, can you recover statistics of the original distribution?

Robust statistics

[Huber], [Tukey] ’60s

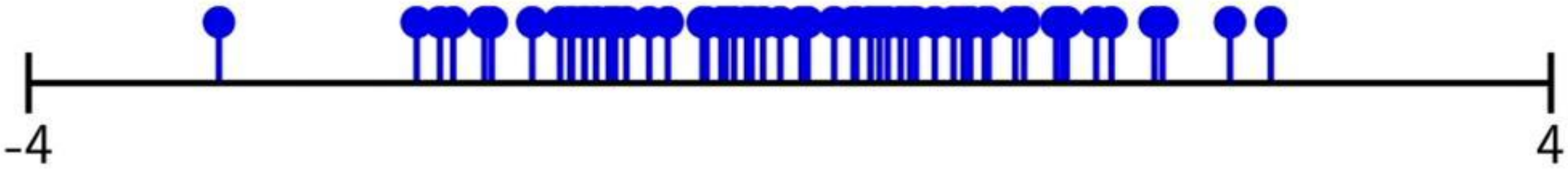
- Given samples from a distribution, where an adversary has moved an ε-fraction of the points arbitrarily, can you recover statistics of the original distribution?



Robust statistics

[Huber], [Tukey] '60s

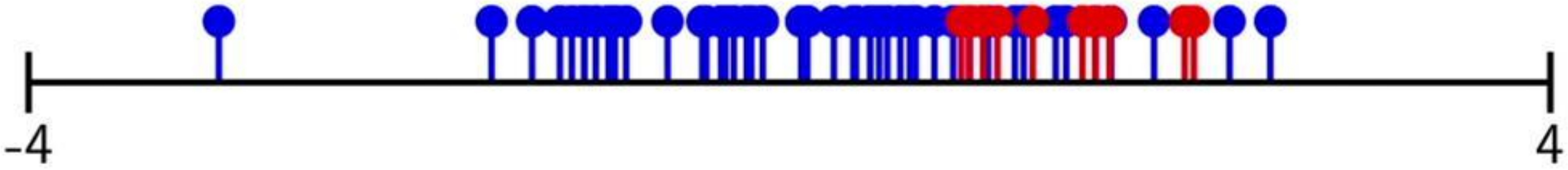
- Given samples from a distribution, where an adversary has moved an ε-fraction of the points arbitrarily, can you recover statistics of the original distribution?



Robust statistics

[Huber], [Tukey] ’60s

- Given samples from a distribution, where an adversary has moved an ε-fraction of the points arbitrarily, can you recover statistics of the original distribution?

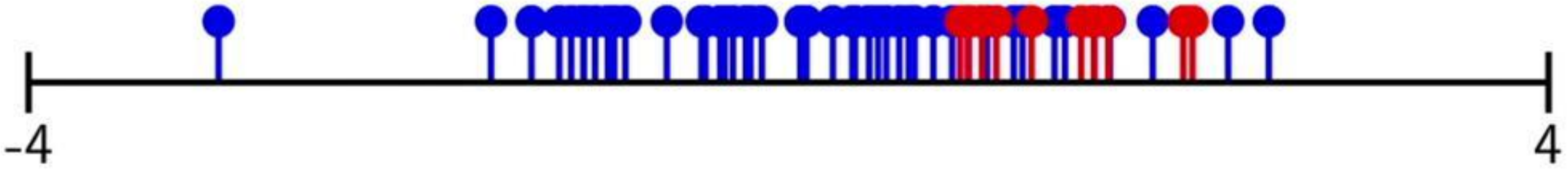


Robust statistics

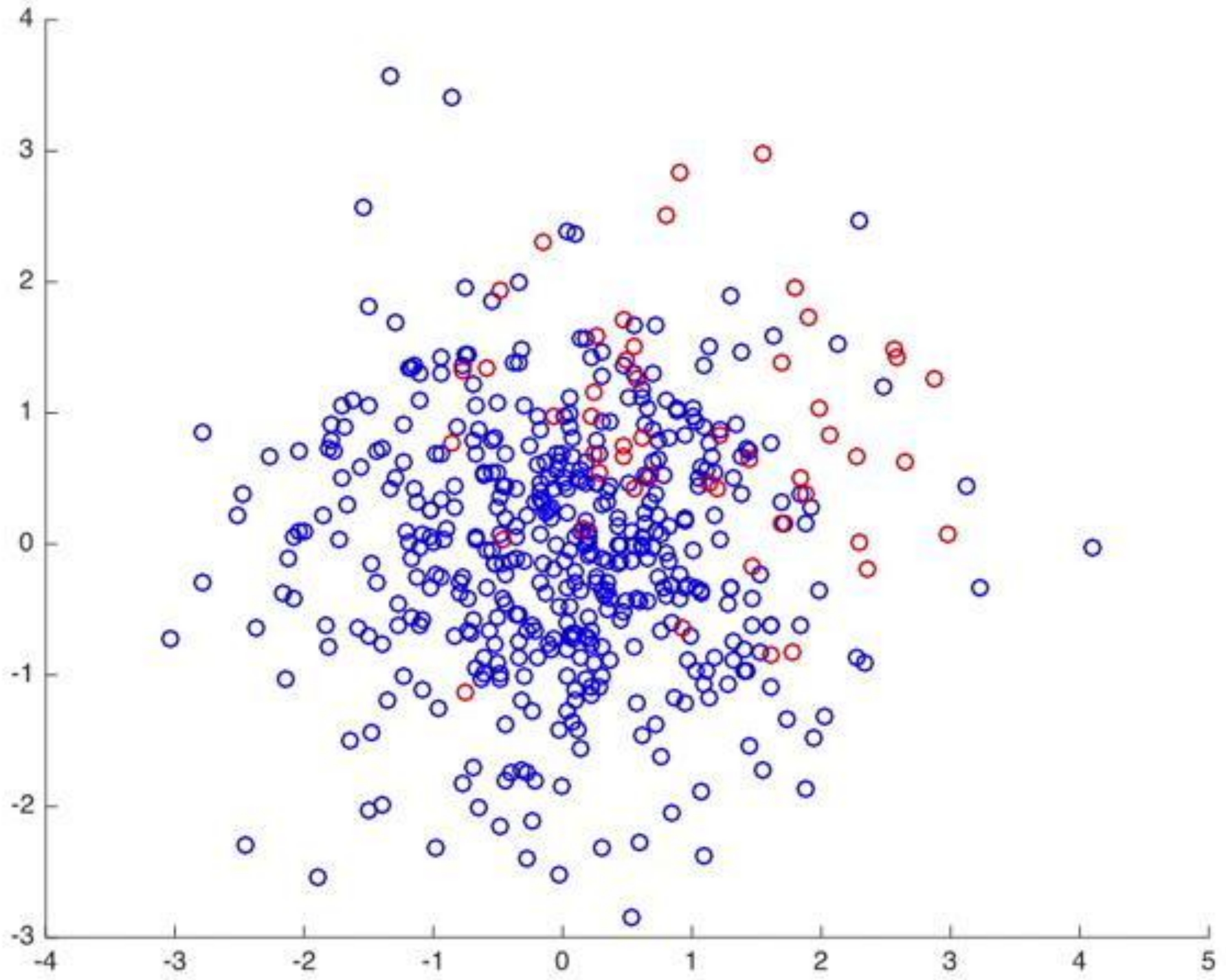
[Huber], [Tukey] '60s

- Given samples from a distribution, where an adversary has moved an ε-fraction of the points arbitrarily, can you recover statistics of the original distribution?

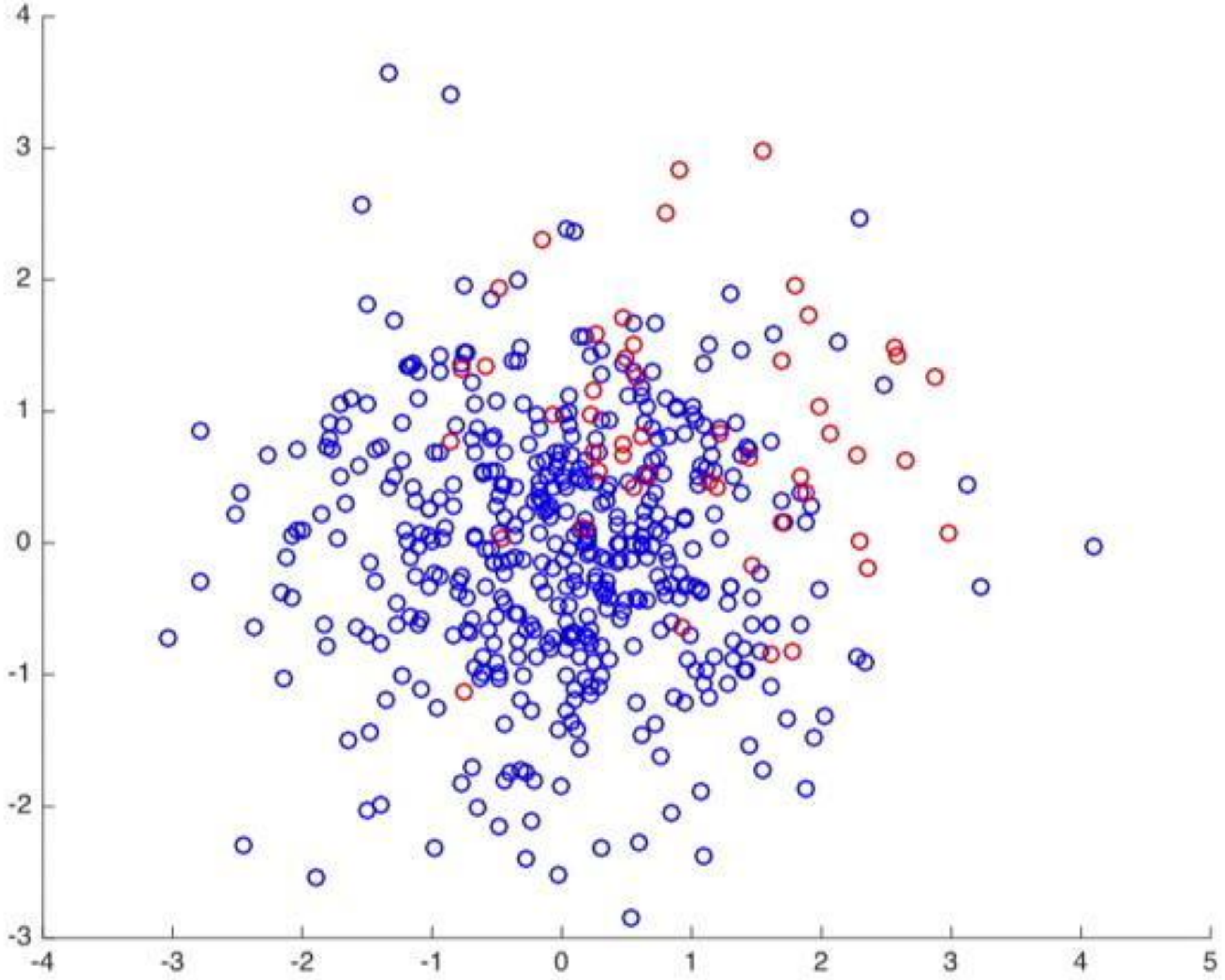
ε-corrupted



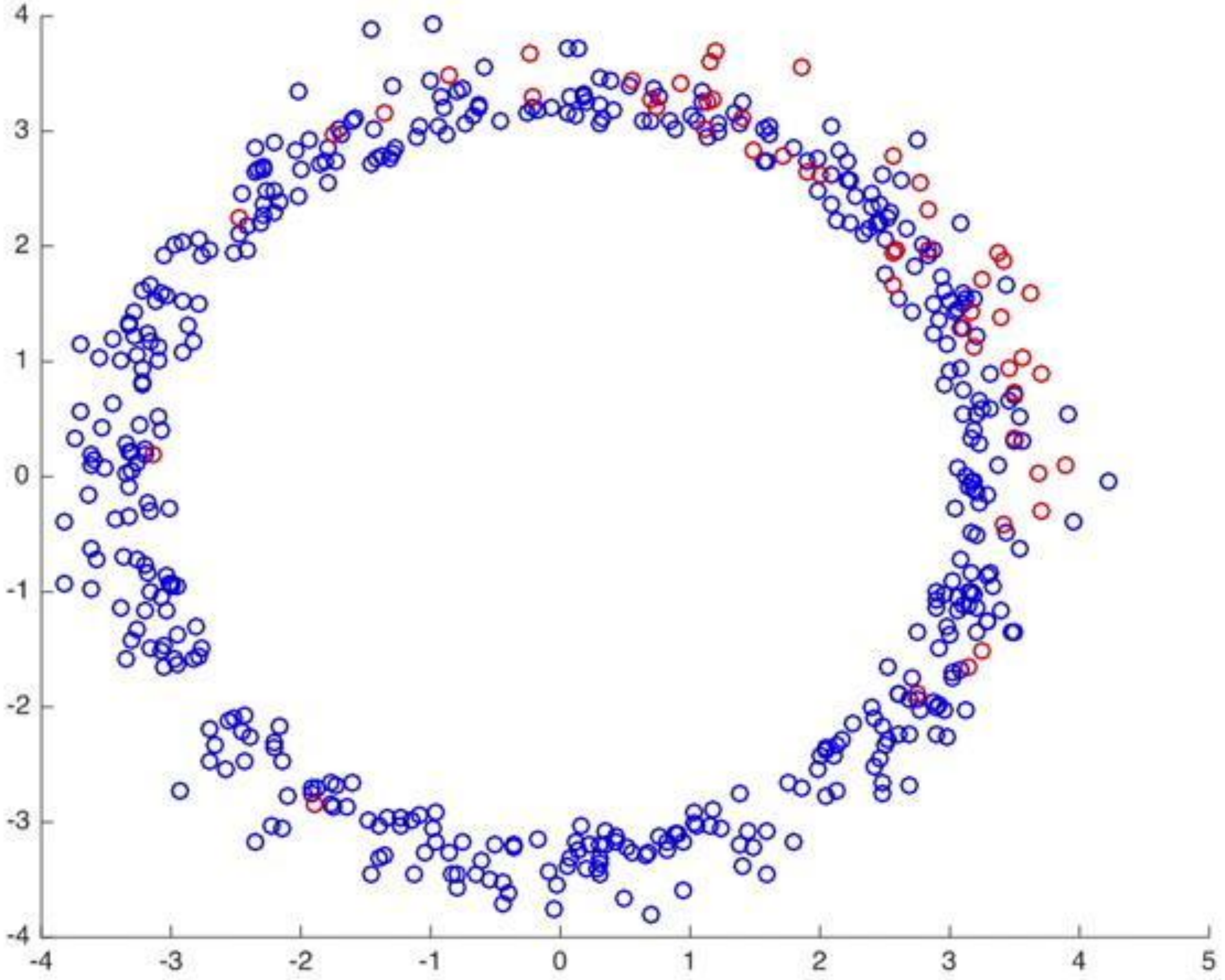
Corruptions in 2 dimensions



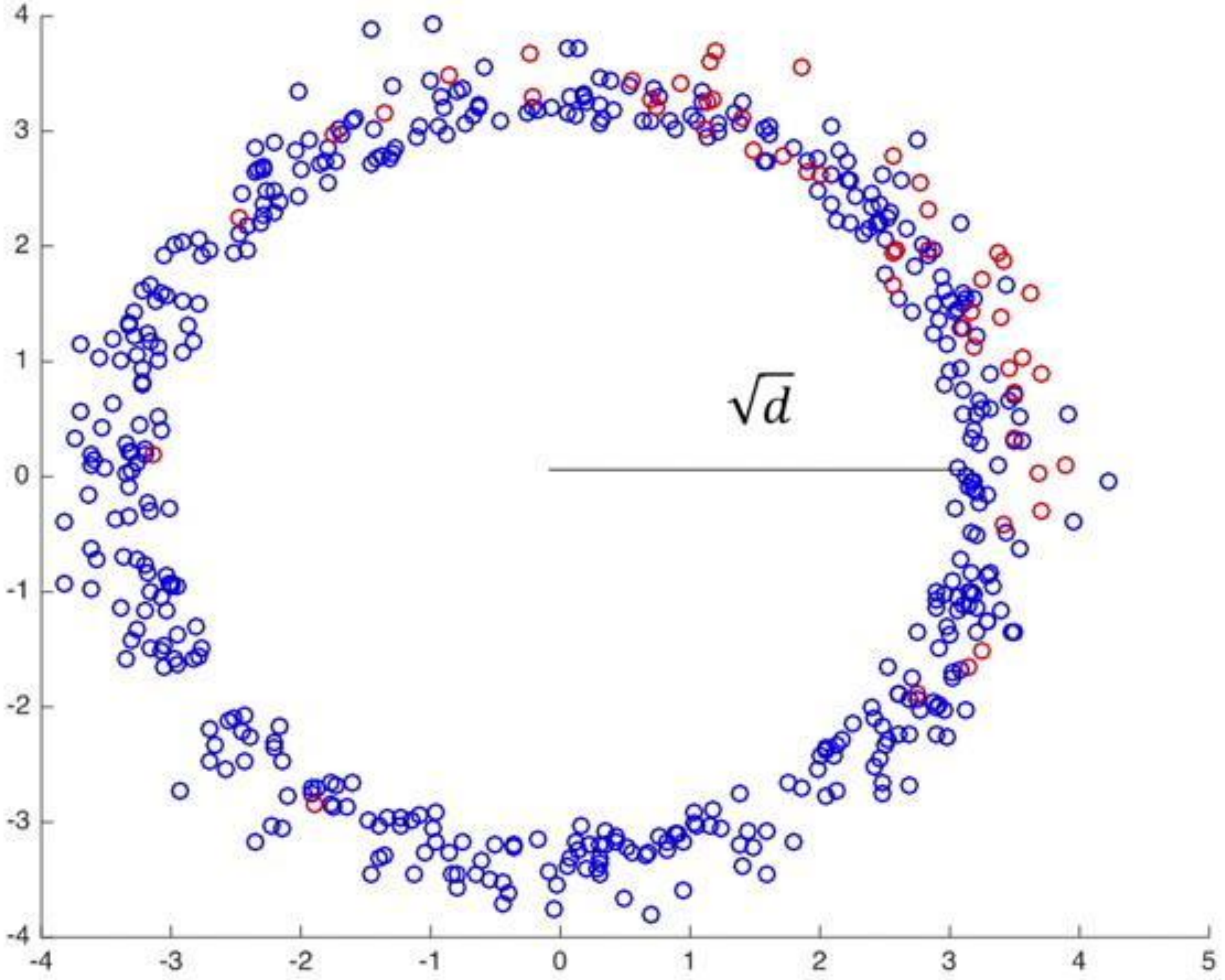
Corruptions in 2 dimensions



Corruptions in high dimensions

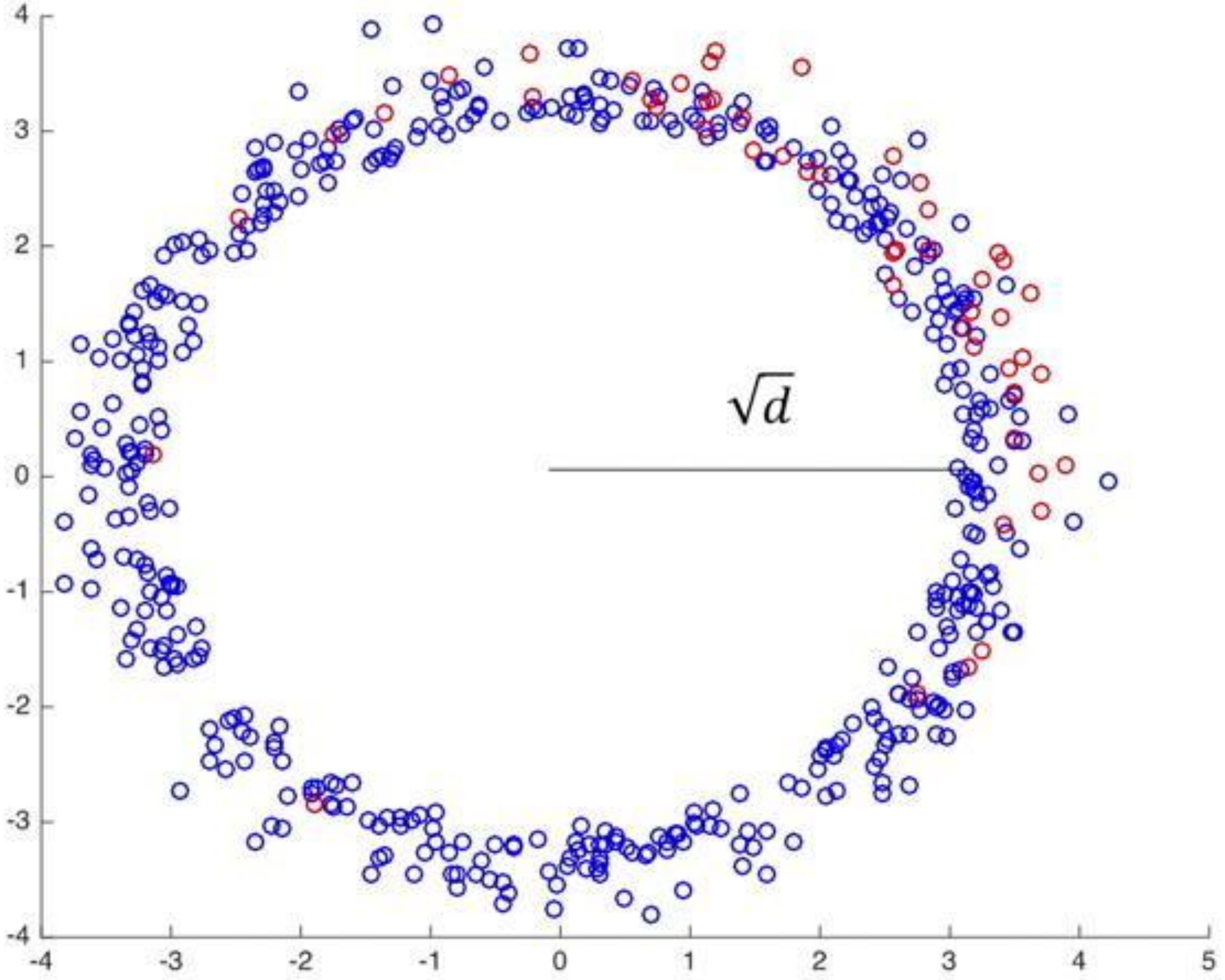


Corruptions in high dimensions



Any method looking for outliers will lose dimension factors

Corruptions in high dimensions



Any method looking for outliers will lose dimension factors

Must look for corruptions globally

A curse of dimensionality?

A curse of dimensionality?

All known approaches for high-dimensional mean estimation either

A curse of dimensionality?

All known approaches for high-dimensional mean estimation either

1. Are computationally intractable in high dimensions; or

A curse of dimensionality?

All known approaches for high-dimensional mean estimation either

1. Are computationally intractable in high dimensions; or
2. Lose accuracy factors which depend polynomially on the dimension

A curse of dimensionality?

All known approaches for high-dimensional mean estimation either

1. Are computationally intractable in high dimensions; or
2. Lose accuracy factors which depend polynomially on the dimension

Is efficient robust estimation possible in high dimensions?

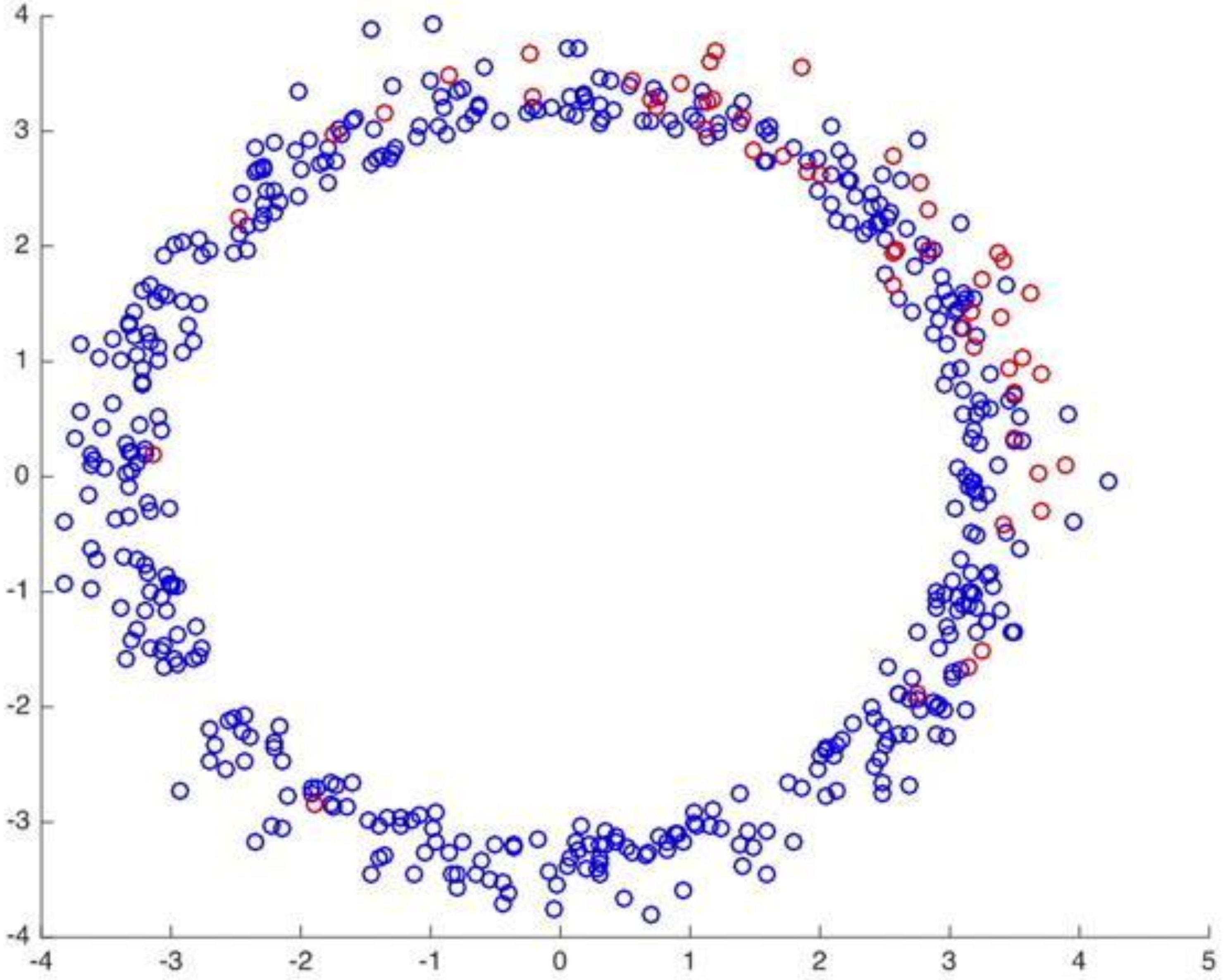
A curse of dimensionality?

All known approaches for high-dimensional mean estimation either

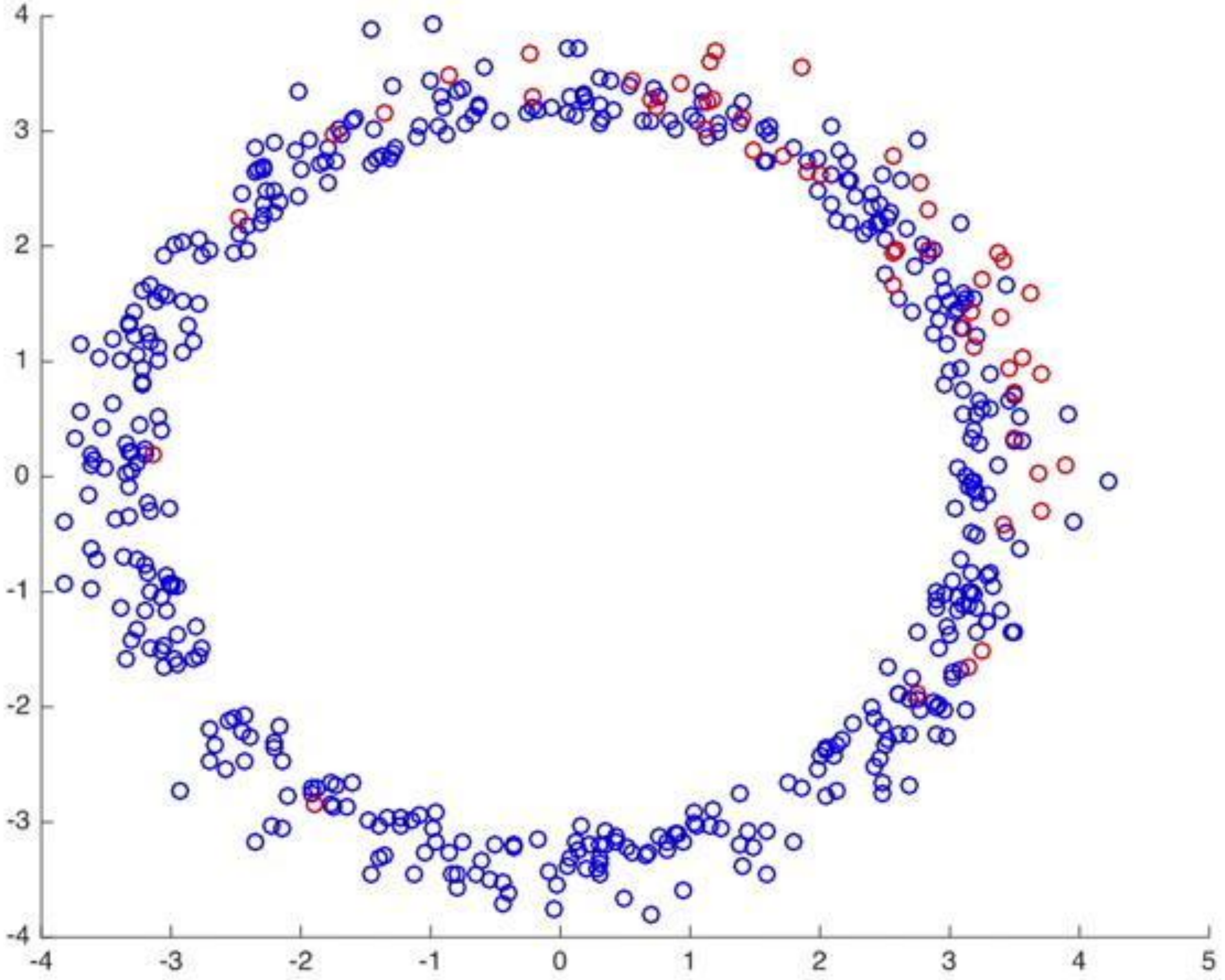
1. Are computationally intractable in high dimensions; or
2. Lose accuracy factors which depend polynomially on the dimension

Is efficient robust estimation possible in high dimensions? **Yes!**

Global corruptions?

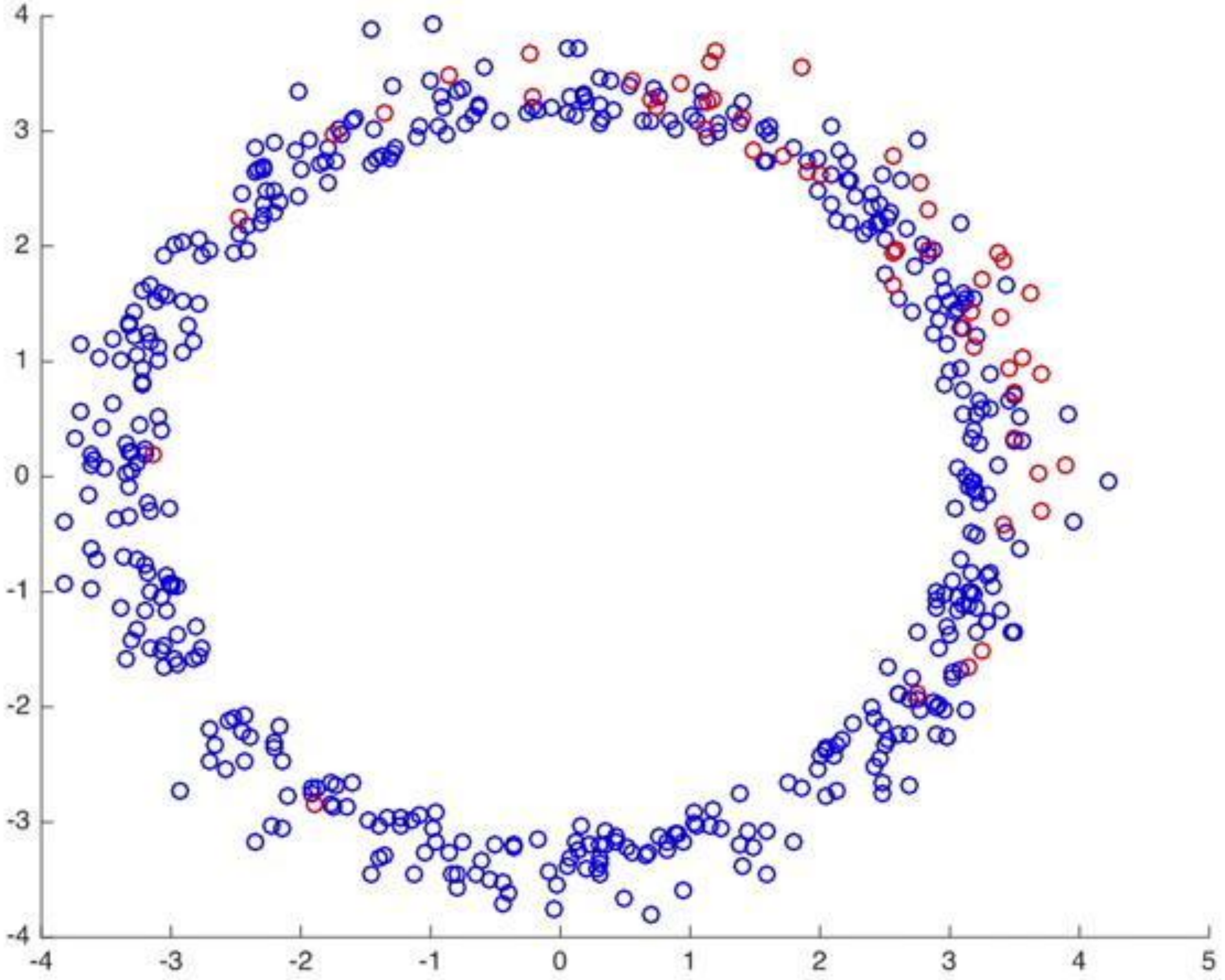


Global corruptions?



Idea: If the corruptions move the mean…

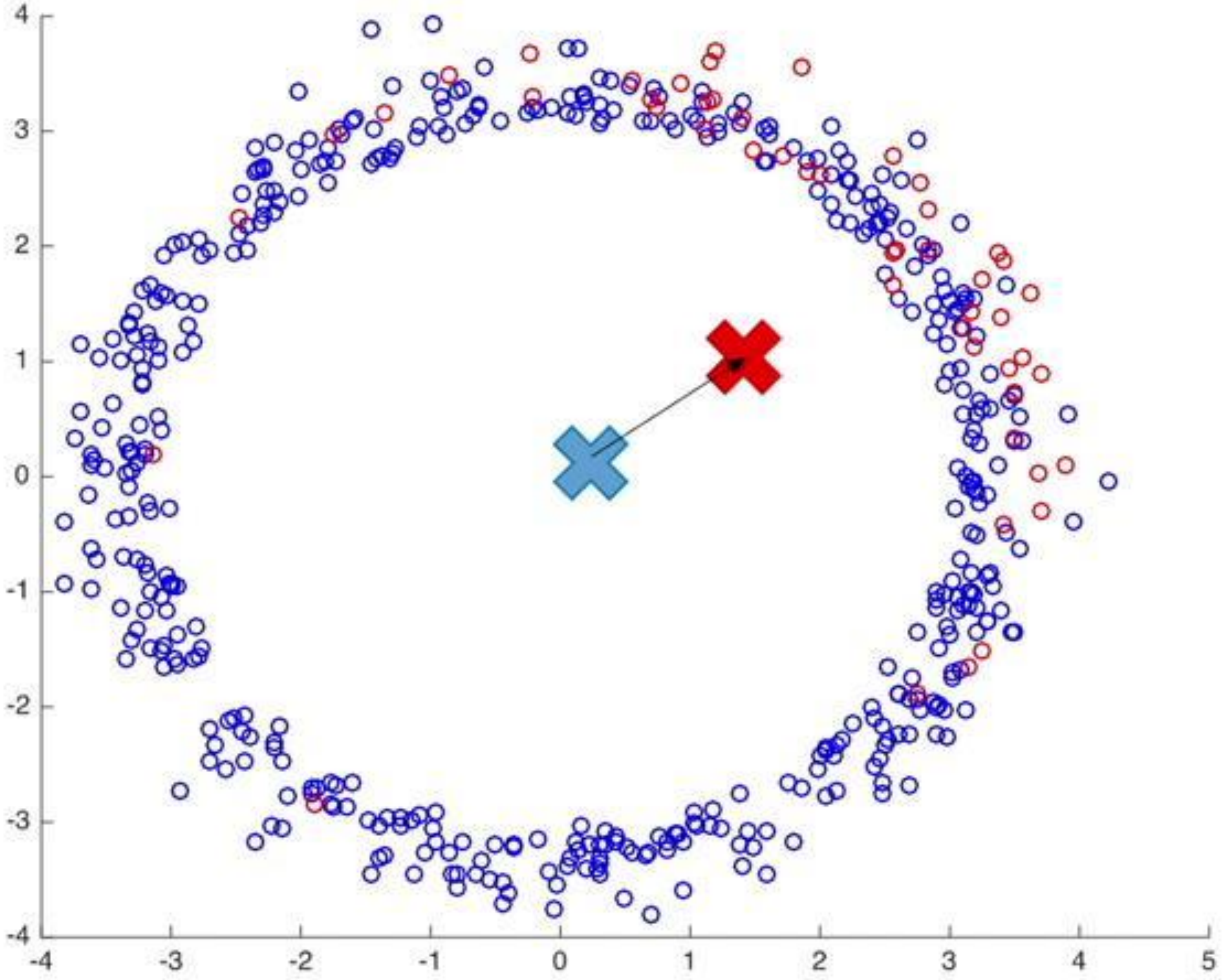
Global corruptions?



Idea: If the corruptions move the mean…

They also shift the covariance matrix!

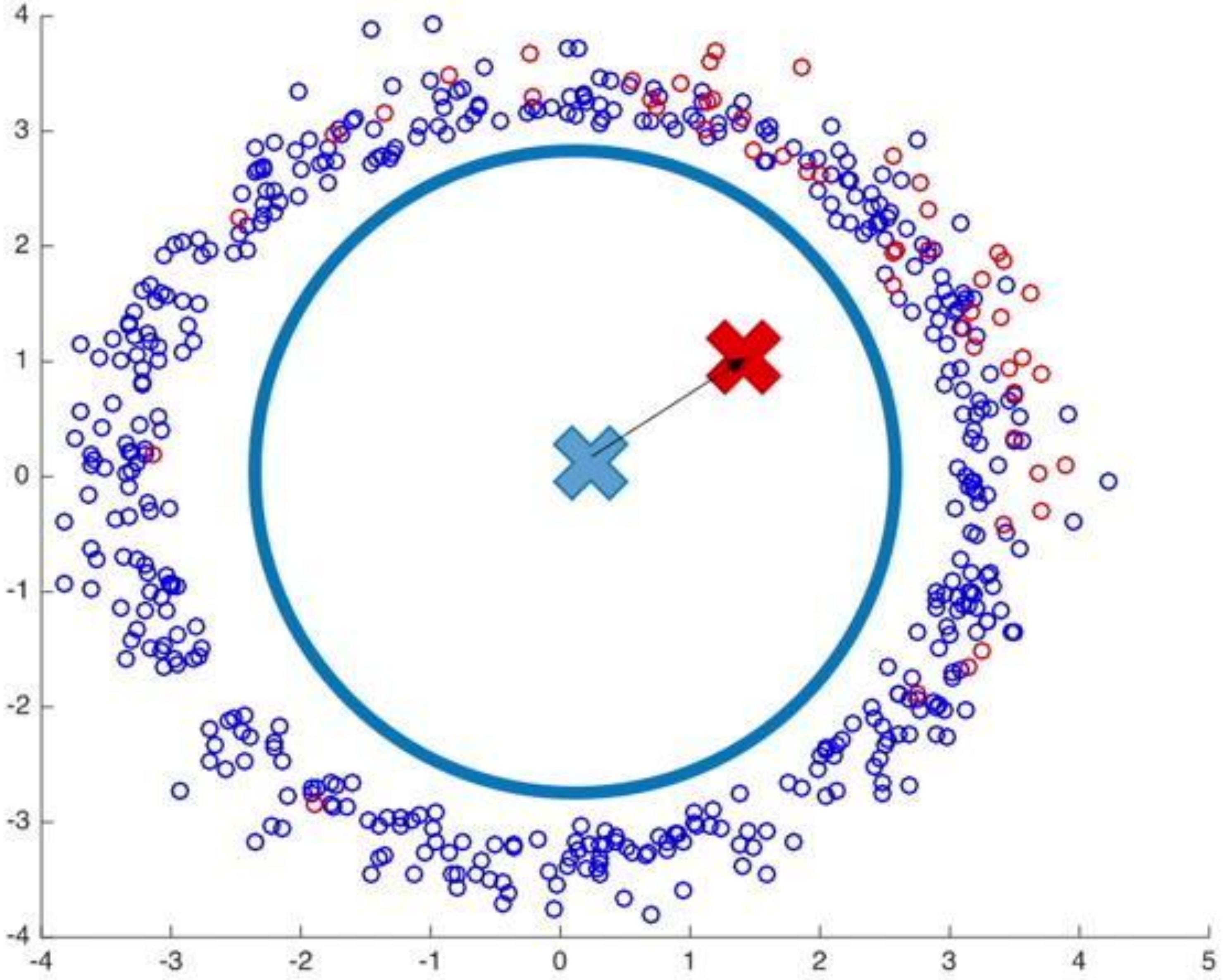
Global corruptions?



Idea: If the corruptions move the mean…

They also shift the covariance matrix!

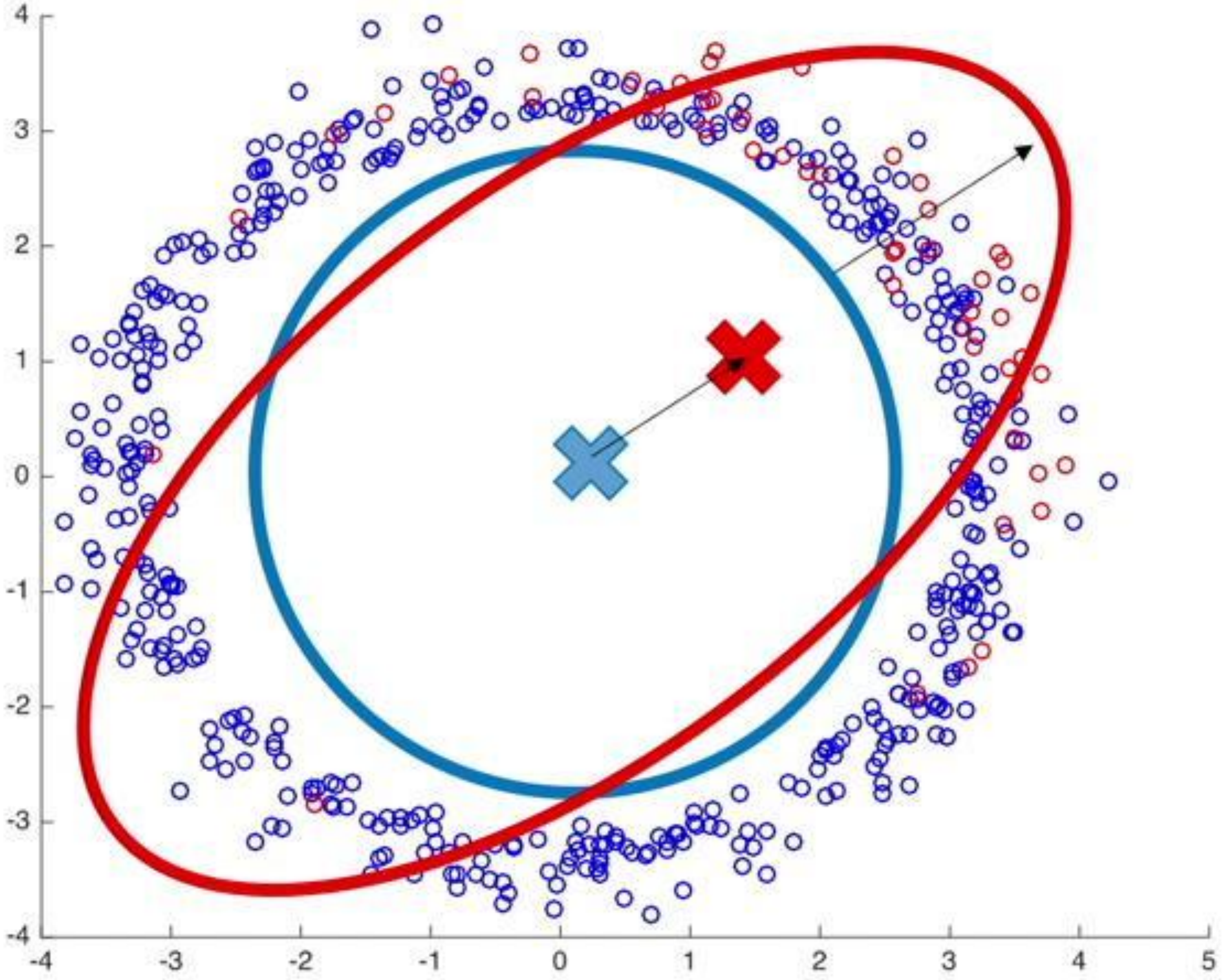
Global corruptions?



Idea: If the corruptions move the mean…

They also shift the covariance matrix!

Global corruptions?



Idea: If the corruptions move the mean…

They also shift the covariance matrix!

Efficient algorithms via spectral signatures

Two consequences of this:

1. If the top eigenvalue of the empirical covariance of your corrupted data is small, then the corruptions aren’t “too bad”.

Efficient algorithms via spectral signatures

Two consequences of this:

1. If the top eigenvalue of the empirical covariance of your corrupted data is small, then the corruptions aren't "too bad".
   - ➢ Can just output the empirical mean!

Efficient algorithms via spectral signatures

Two consequences of this:

1. If the top eigenvalue of the empirical covariance of your corrupted data is small, then the corruptions aren't "too bad".
   - Can just output the empirical mean!

2. If the top eigenvalue is large, then it can only be large because the bad points are too big in this direction.

Efficient algorithms via spectral signatures

Two consequences of this:

1. If the top eigenvalue of the empirical covariance of your corrupted data is small, then the corruptions aren't "too bad".
   - Can just output the empirical mean!

2. If the top eigenvalue is large, then it can only be large because the bad points are too big in this direction.
   - The top eigenvector gives a direction where the bad points are prominent!

Filtering: A Simple Meta-Algorithm

Given corrupted dataset S

Filtering: A Simple Meta-Algorithm

Given corrupted dataset S

- Let $\hat{\mu}$ be the empirical mean of S

Filtering: A Simple Meta-Algorithm

Given corrupted dataset S

- Let $\hat{\mu}$ be the empirical mean of S
- Let $\hat{\Sigma}$ be the empirical covariance of S

Filtering: A Simple Meta-Algorithm

Given corrupted dataset S

- Let $\hat{\mu}$ be the empirical mean of S
- Let $\hat{\Sigma}$ be the empirical covariance of S
- $(\lambda, v) \leftarrow$ top eigenvalue/vector of $\hat{\Sigma}$

Filtering: A Simple Meta-Algorithm

Given corrupted dataset S

- Let $\hat{\mu}$ be the empirical mean of S
- Let $\hat{\Sigma}$ be the empirical covariance of S
- $(\lambda, v) \leftarrow$ top eigenvalue/vector of $\hat{\Sigma}$
- If λ is not too large

Filtering: A Simple Meta-Algorithm

Given corrupted dataset S

- Let $\hat{\mu}$ be the empirical mean of S
- Let $\hat{\Sigma}$ be the empirical covariance of S
- $(\lambda, v) \leftarrow$ top eigenvalue/vector of $\hat{\Sigma}$
- If λ is not too large
  - Output $\hat{\mu}$

Filtering: A Simple Meta-Algorithm

Given corrupted dataset S

- Let $\hat{\mu}$ be the empirical mean of S
- Let $\hat{\Sigma}$ be the empirical covariance of S
- $(\lambda, v) \leftarrow$ top eigenvalue/vector of $\hat{\Sigma}$
- If λ is not too large
    - Output $\hat{\mu}$
- Otherwise,

Filtering: A Simple Meta-Algorithm

Given corrupted dataset S

- Let $\hat{\mu}$ be the empirical mean of S
- Let $\hat{\Sigma}$ be the empirical covariance of S
- $(\lambda, v) \leftarrow$ top eigenvalue/vector of $\hat{\Sigma}$
- If λ is not too large
  - Output $\hat{\mu}$
- Otherwise,
  - Project the data points in the direction of v

Filtering: A Simple Meta-Algorithm

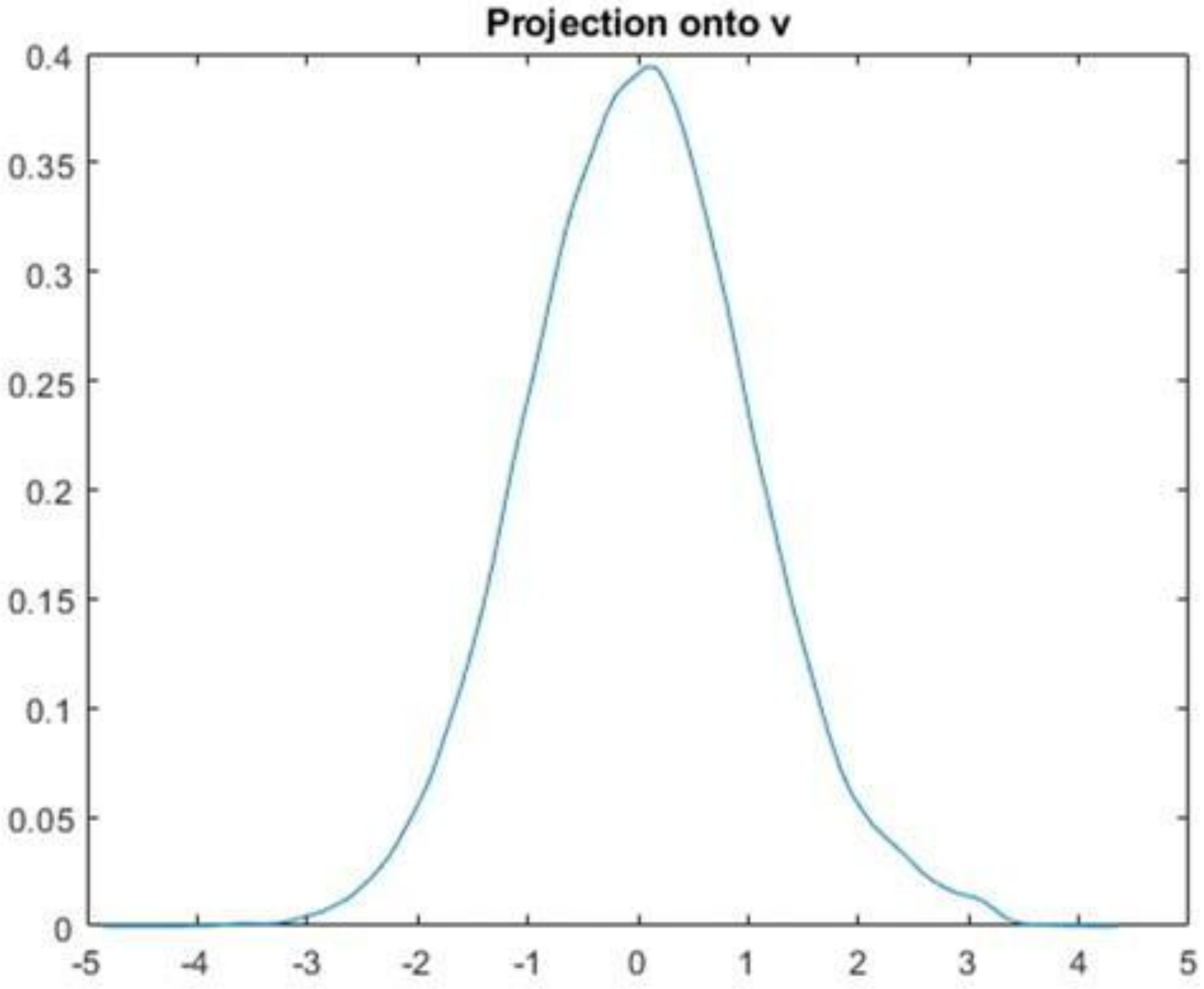
Given corrupted dataset S

- Let $\hat{\mu}$ be the empirical mean of S
- Let $\hat{\Sigma}$ be the empirical covariance of S
- $(\lambda, v) \leftarrow$ top eigenvalue/vector of $\hat{\Sigma}$
- If λ is not too large
  - Output $\hat{\mu}$
- Otherwise,
  - Project the data points in the direction of v
  - Remove (or downweight) the largest data points in this direction

Filtering: A Simple Meta-Algorithm

Given corrupted dataset S

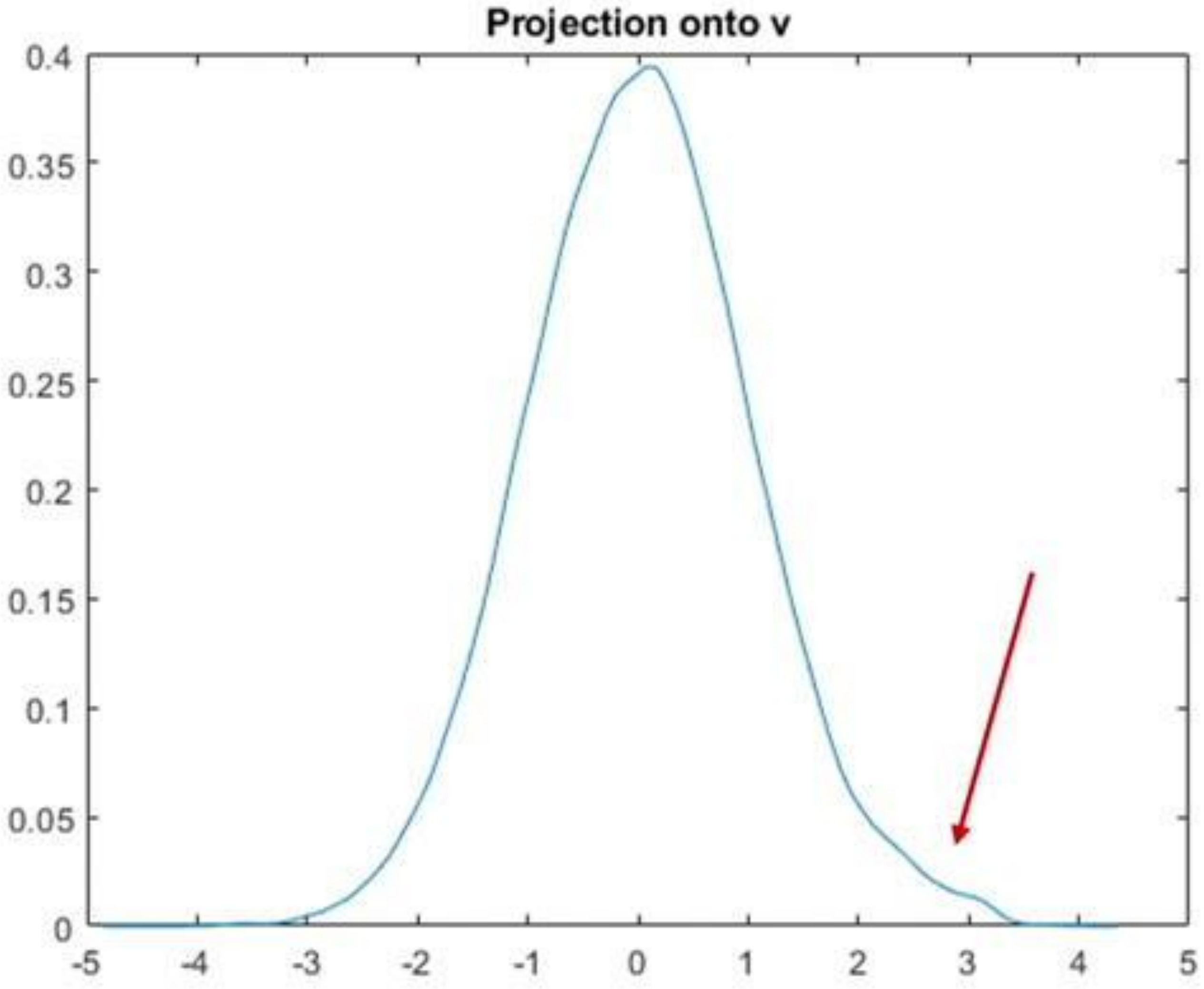
- Let $\hat{\mu}$ be the empirical mean of S
- Let $\hat{\Sigma}$ be the empirical covariance of S
- $(\lambda, v) \leftarrow$ top eigenvalue/vector of $\hat{\Sigma}$
- If λ is not too large
  - Output $\hat{\mu}$
- Otherwise,
  - Project the data points in the direction of v
  - Remove (or downweight) the largest data points in this direction



Filtering: A Simple Meta-Algorithm

Given corrupted dataset S

- Let $\hat{\mu}$ be the empirical mean of S
- Let $\hat{\Sigma}$ be the empirical covariance of S
- $(\lambda, v) \leftarrow$ top eigenvalue/vector of $\hat{\Sigma}$
- If λ is not too large
  - Output $\hat{\mu}$
- Otherwise,
  - Project the data points in the direction of v
  - Remove (or downweight) the largest data points in this direction



A single iteration runs in nearly linear time!

Our Results

Given an ε-corrupted set of samples that is sufficiently large from...	...we can efficiently get an estimate of the true mean to ℓ_2 error:
a distribution with bounded second moment	$O(\sqrt{\varepsilon})$ [LRV16, DKKLMS16, DKKLMS17]

Our Results

Given an ε-corrupted set of samples that is sufficiently large from...	...we can efficiently get an estimate of the true mean to ℓ_2 error:
a distribution with bounded second moment	$O(\sqrt{\varepsilon})$ [LRV16, DKKLMS16, DKKLMS17]
a Gaussian (or sub-Gaussian distribution) with identity covariance	$O(\varepsilon\sqrt{\log 1/\varepsilon})$ [DKKLMS17,SCV17]

Our Results

Given an ε-corrupted set of samples that is sufficiently large from...	...we can efficiently get an estimate of the true mean to ℓ_2 error:
a distribution with bounded second moment	$O(\sqrt{\varepsilon})$ [LRV16, DKKLMS16, DKKLMS17]
a Gaussian (or sub-Gaussian distribution) with identity covariance	$O(\varepsilon\sqrt{\log 1/\varepsilon}))$ [DKKLMS17,SCV17]
a Gaussian with unknown covariance	$O(\varepsilon \log 1/\varepsilon)$ [DKKLMS16]

Our Results

Given an ε-corrupted set of samples that is sufficiently large from...	...we can efficiently get an estimate of the true mean to ℓ_2 error:
a distribution with bounded second moment	$O(\sqrt{\varepsilon})$ [LRV16, DKKLMS16, DKKLMS17]
a Gaussian (or sub-Gaussian distribution) with identity covariance	$O(\varepsilon\sqrt{\log 1/\varepsilon})$ [DKKLMS17,SCV17]
a Gaussian with unknown covariance	$O(\varepsilon \log 1/\varepsilon)$ [DKKLMS16]
a "nice" distribution with bounded t-th moments	$O(\varepsilon^{1-1/t})$ [HL18, KS18]

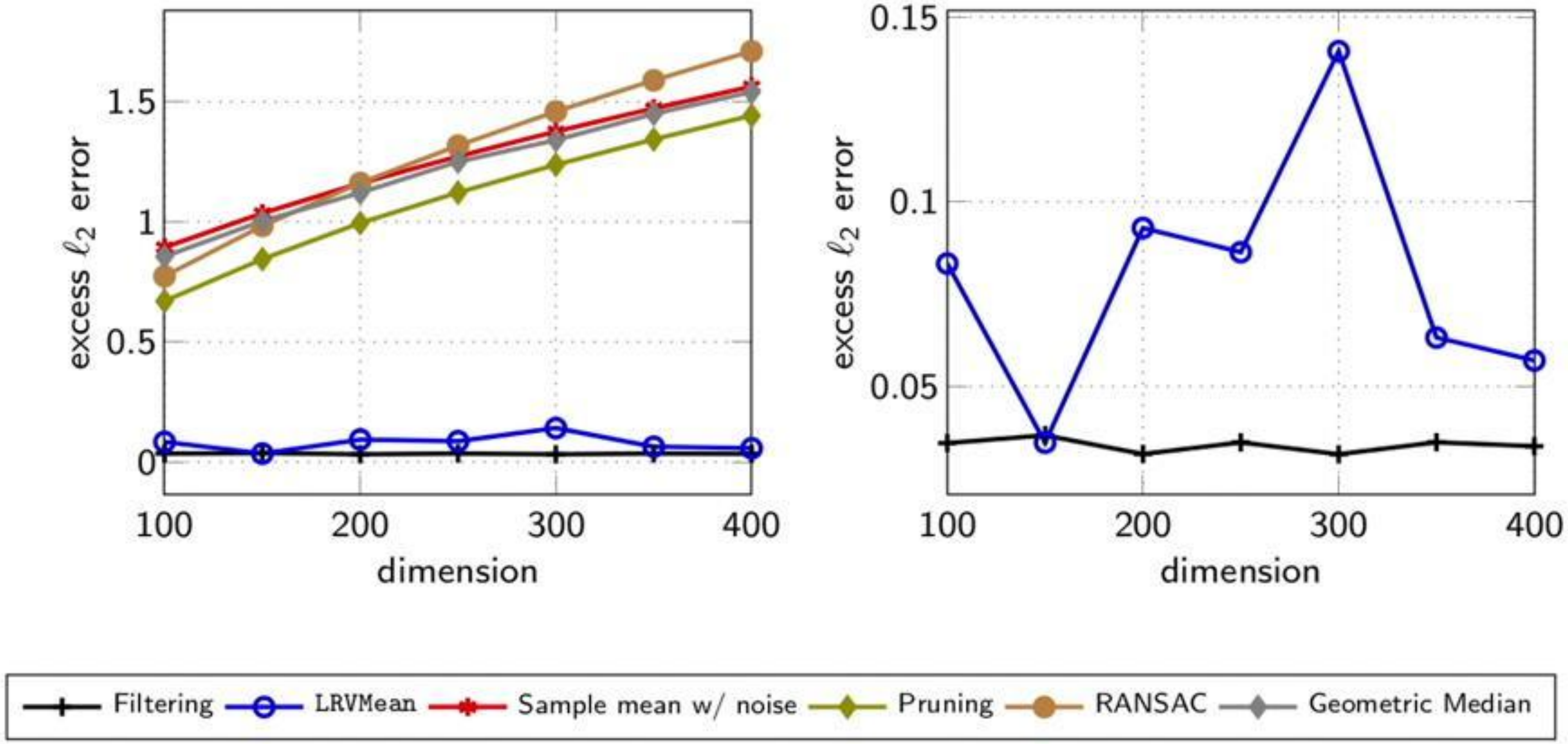
Our Results

Given an ε-corrupted set of samples that is sufficiently large from...	...we can efficiently get an estimate of the true mean to ℓ_2 error:
a distribution with bounded second moment	$O(\sqrt{\varepsilon})$ [LRV16, DKKLMS16, DKKLMS17]
a Gaussian (or sub-Gaussian distribution) with identity covariance	$O(\varepsilon\sqrt{\log 1/\varepsilon}))$ [DKKLMS17,SCV17]
a Gaussian with unknown covariance	$O(\varepsilon \log 1/\varepsilon)$ [DKKLMS16]
a "nice" distribution with bounded t-th moments	$O(\varepsilon^{1-1/t})$ [HL18, KS18]

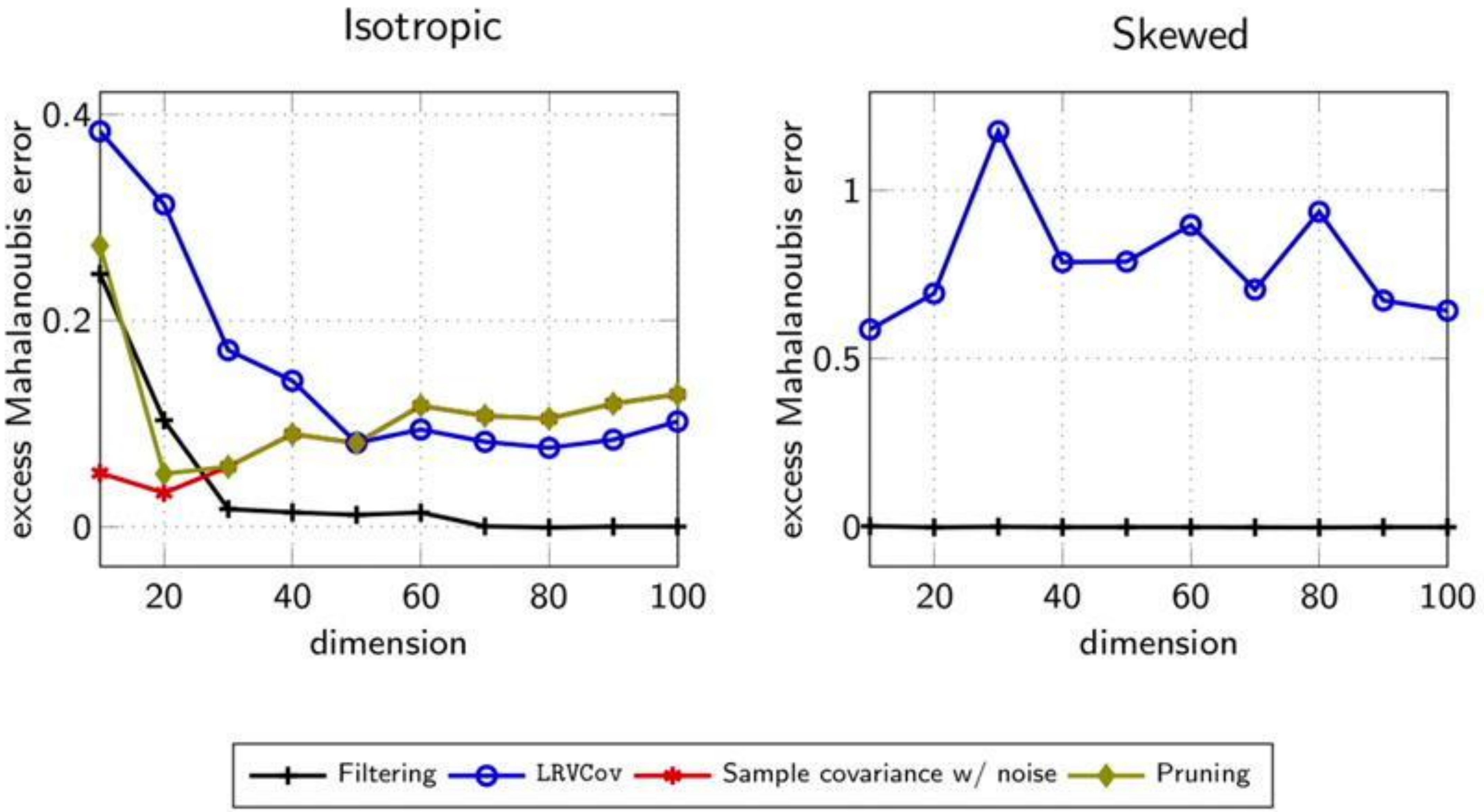
For all cases, these are the first efficient dimension-independent guarantees!

Also sparsity [L17, DBS17], list learning [CSV17, MV17], graphical models [DKS18], general norms [SCV17], federated learning [QV17], sparse regression [KKM18, CLL19] etc...

Synthetic Experiments, Unknown Mean

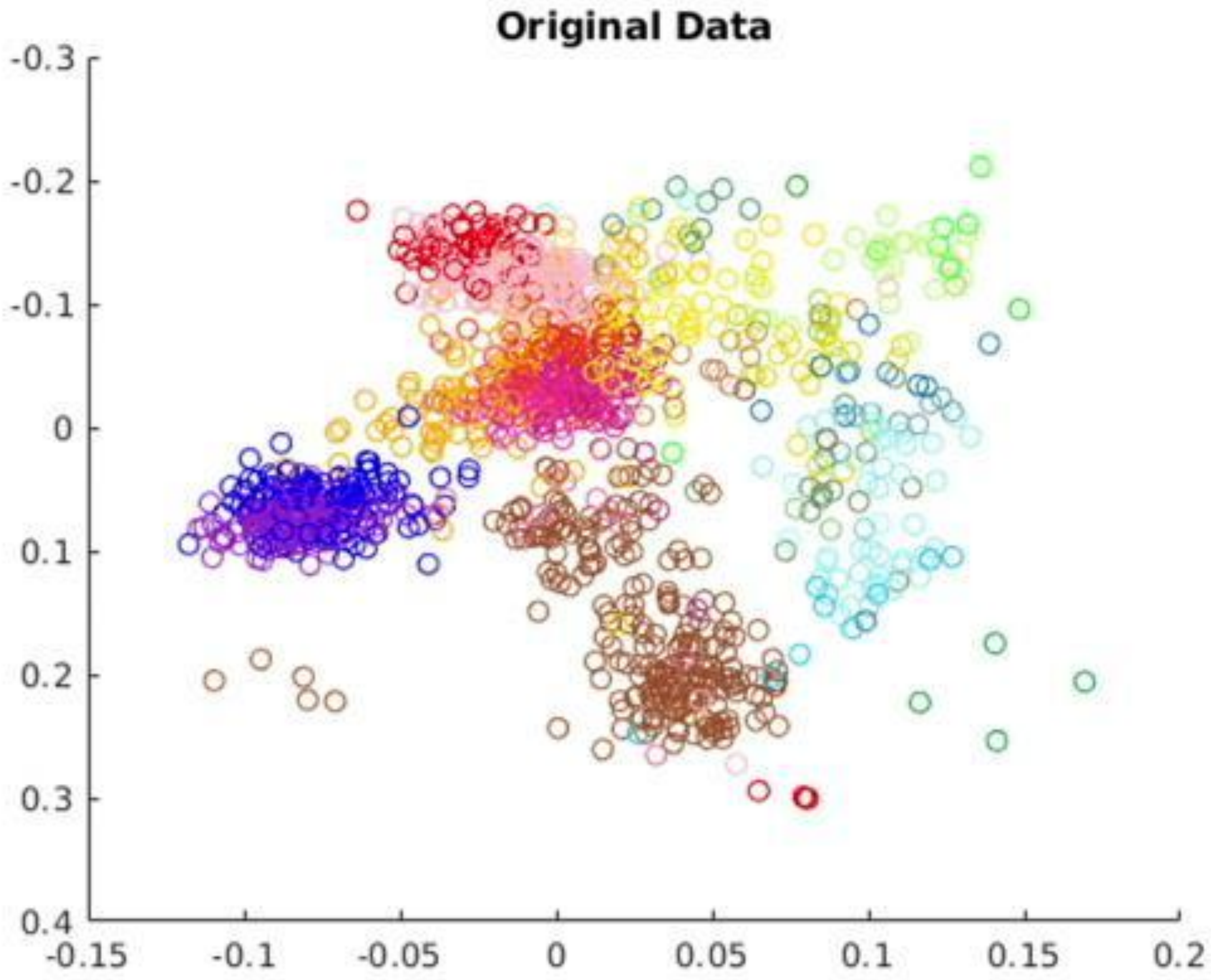


Synthetic Experiments, Unknown Covariance



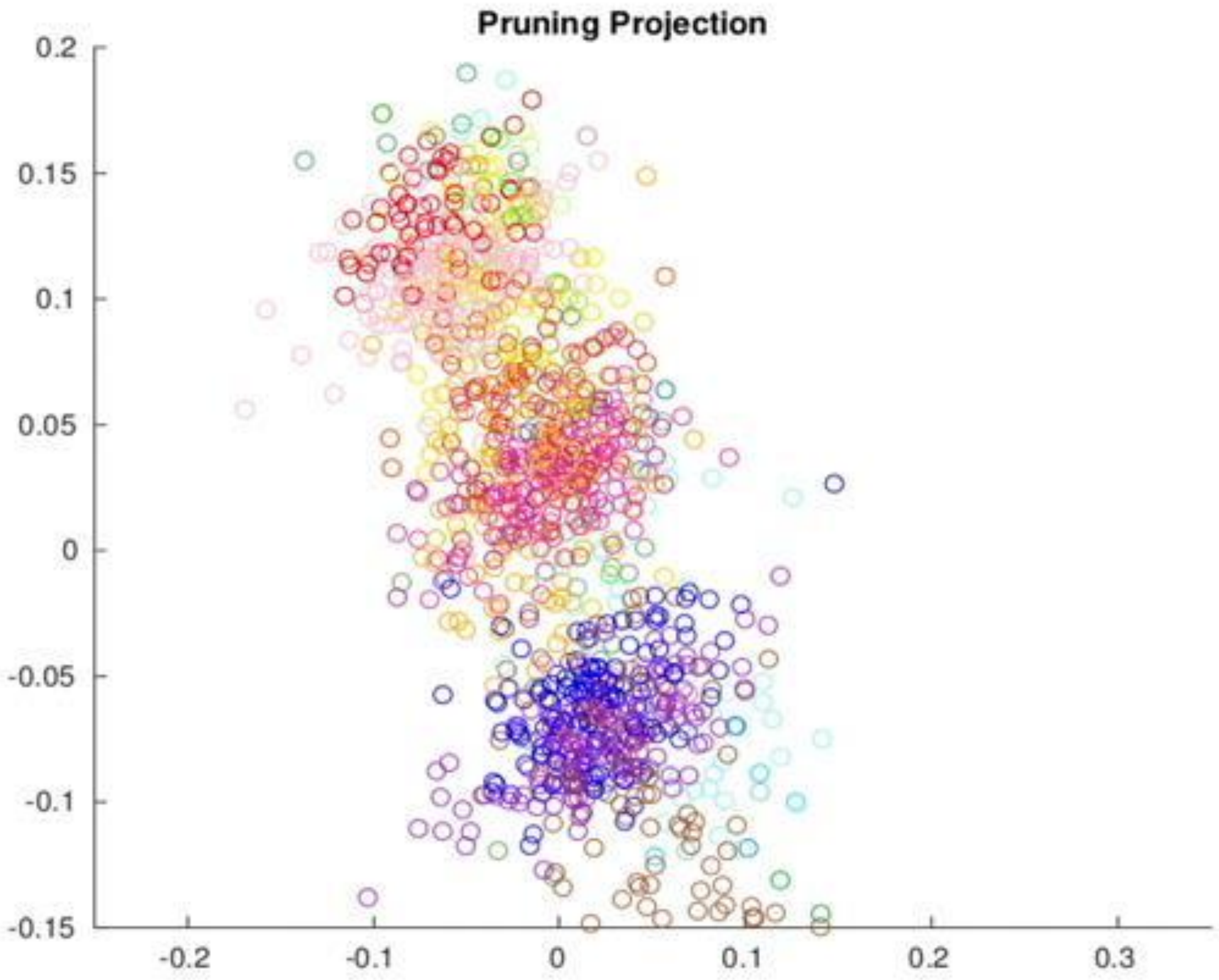
Gene Expression PCA Contains Europe

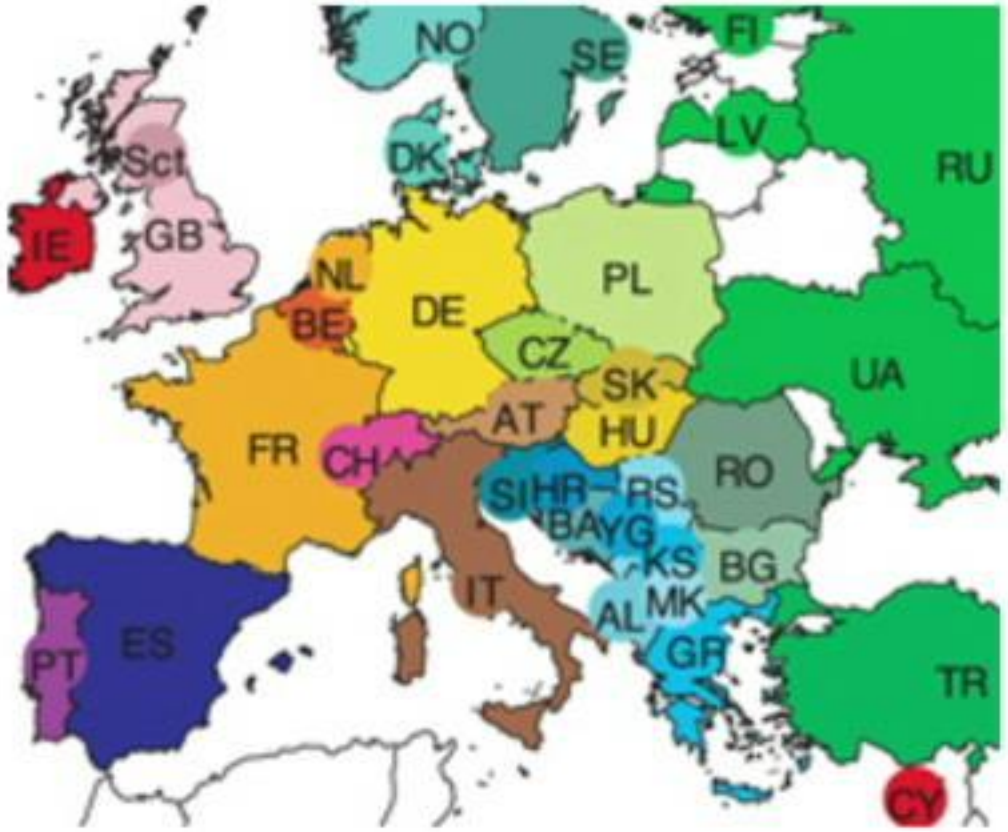
- Genes Mirror Geography in Europe. [Novembre et al.], *Nature* ‘08



Naively, Corruptions Destroy Europe

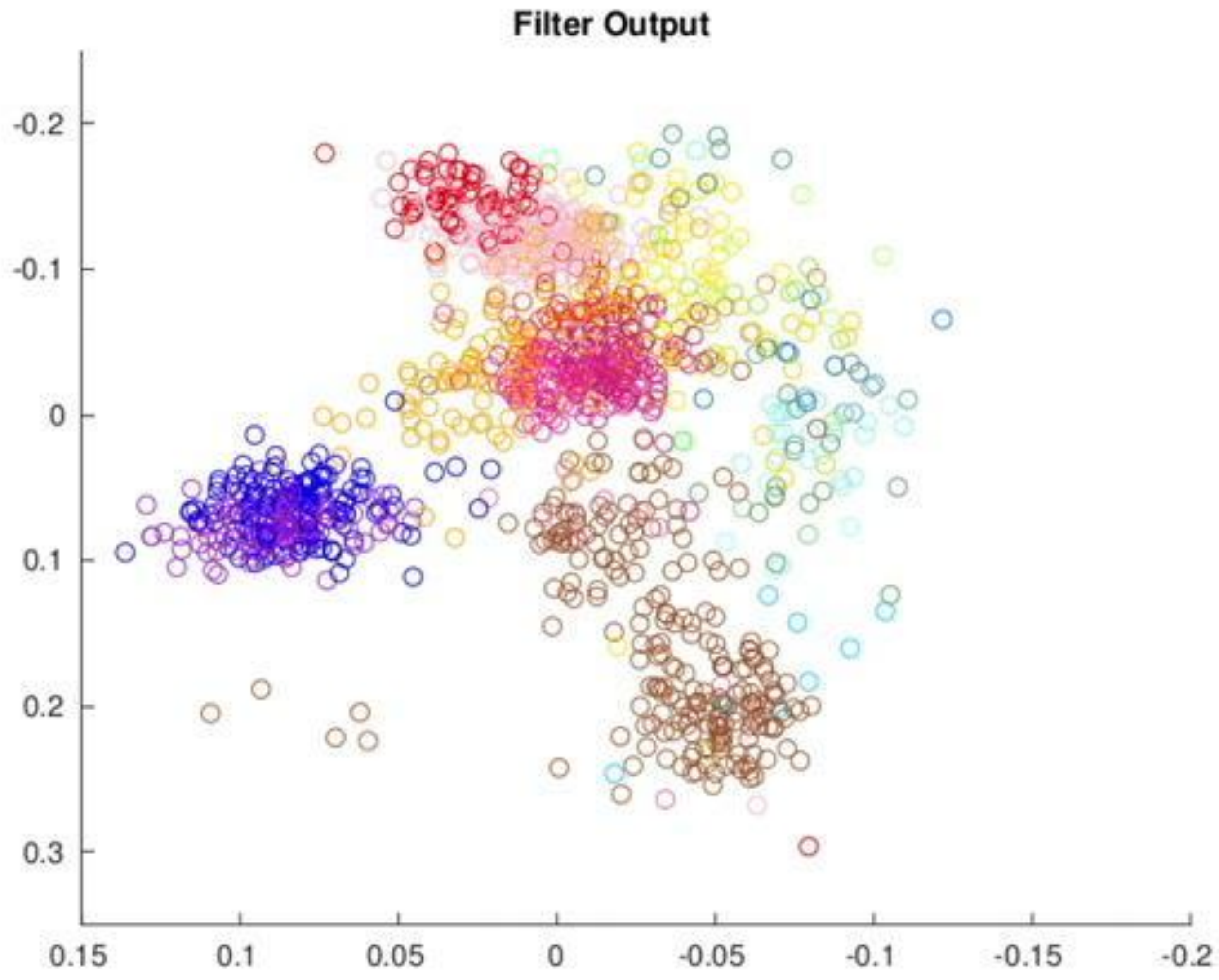
- Genes Mirror Geography in Europe. [Novembre et al.'08]

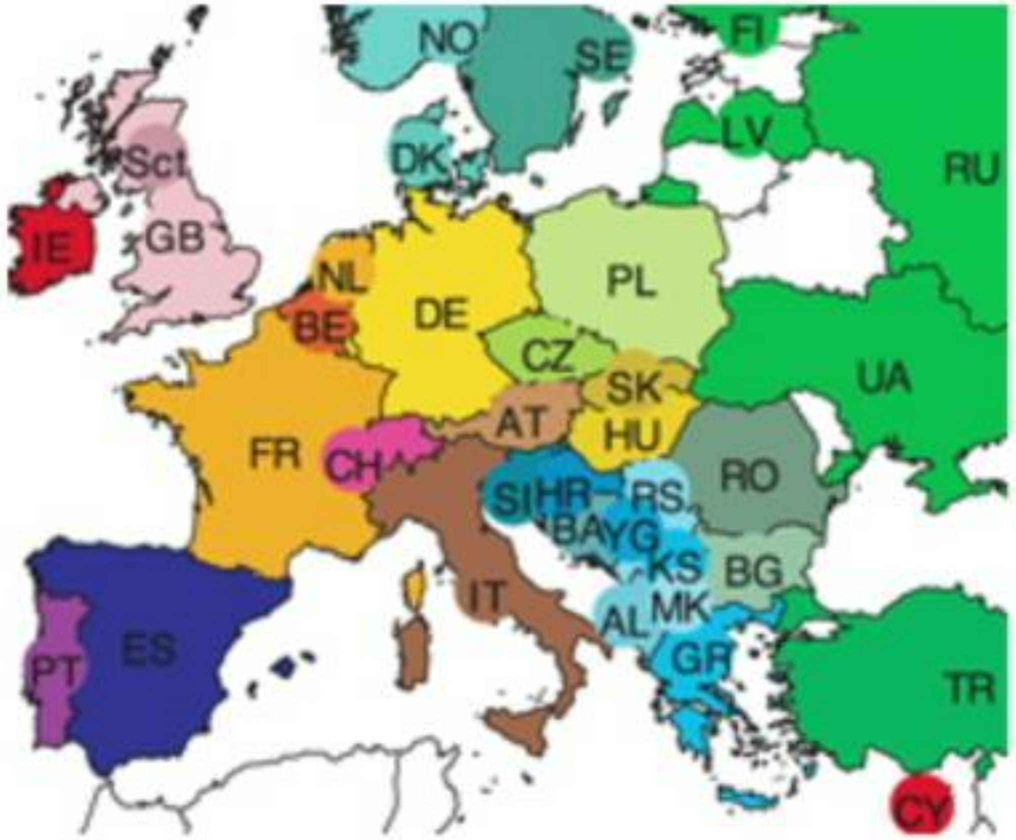




Our Algorithms Fix Europe!

- Genes Mirror Geography in Europe. [Novembre et al.'08]





Application to outlier detection

Application to outlier detection

The filter also directly gives us scores which rank how suspicious each data point is.

Application to outlier detection

The filter also directly gives us scores which rank how suspicious each data point is.

We can directly use this as a method for outlier detection.

Application to outlier detection

The filter also directly gives us scores which rank how suspicious each data point is.

We can directly use this as a method for outlier detection.

Recent work of [Dong, Hopkins, L] give a more sophisticated score motivated by robust outlier detection called **quantum entropy (QUE) scoring**

Application to outlier detection

The filter also directly gives us scores which rank how suspicious each data point is.

We can directly use this as a method for outlier detection.

Recent work of [Dong, Hopkins, L] give a more sophisticated score motivated by robust outlier detection called **quantum entropy (QUE) scoring**

QUE scores outperform previous SOTA on both synthetic and real world outlier detection tasks!

Experimental setup (synthetic)

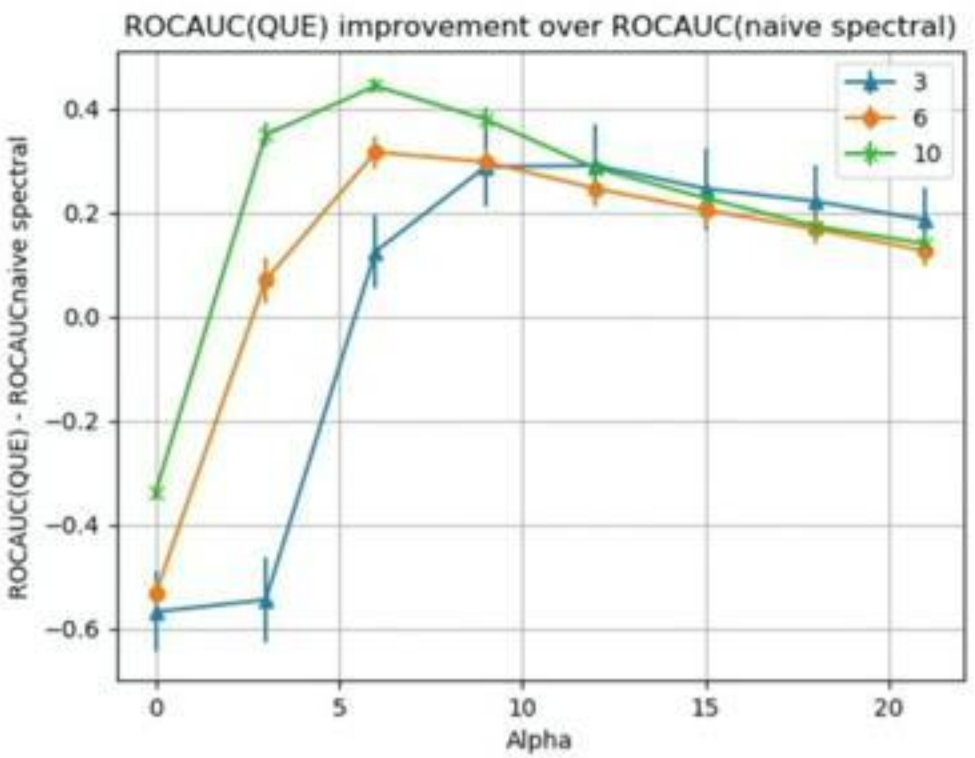
[Dong, Hopkins, L], to appear, NeurIPS 2020

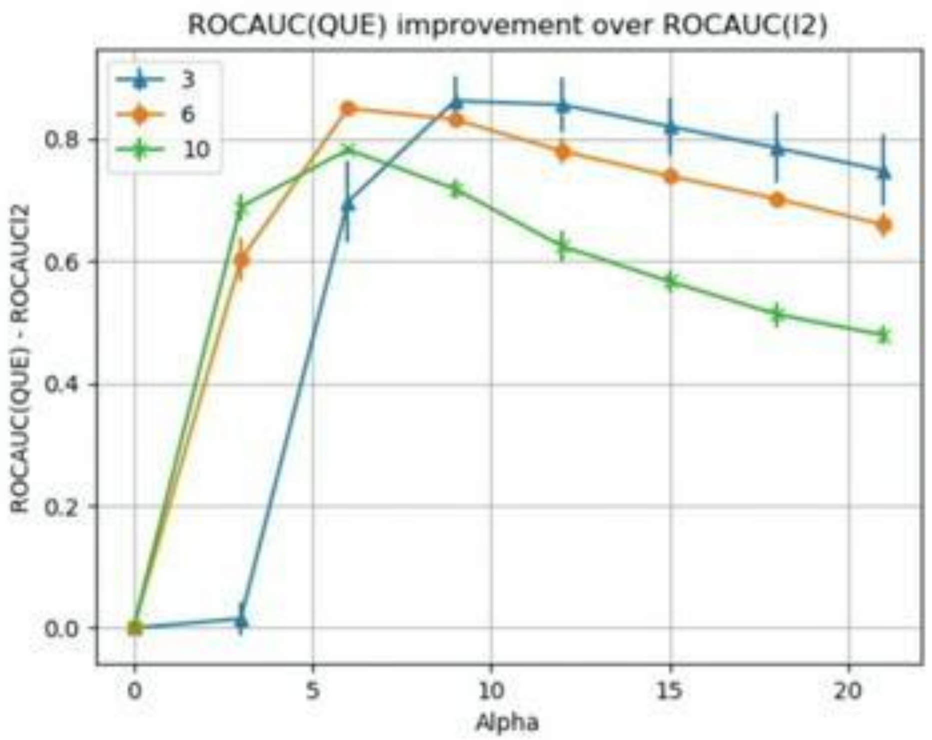
- Inliers are Gaussian data, outliers are in k roughly orthogonal directions

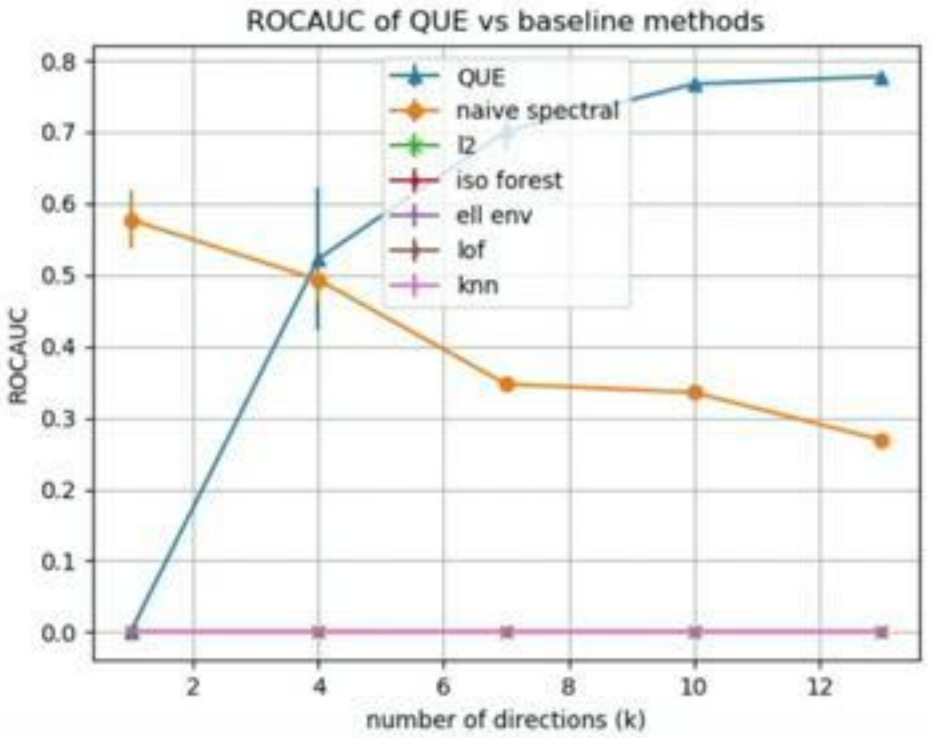
Experimental setup (synthetic)

[Dong, Hopkins, L], to appear, NeurIPS 2020

- Inliers are Gaussian data, outliers are in k roughly orthogonal directions







Experimental setup (CIFAR-10)

[Dong, Hopkins, L], to appear, NeurIPS 2020

- Inliers are images CIFAR-10, outliers are images from CIFAR-10 grouped into k groups, where each group has some set of "dead" pixels
- We whiten the data using another set of uncorrupted images from CIFAR-10.

Beyond robust statistics

Can we “robust-ify” more complicated objectives, like supervised learning?

e.g. regression, SVM

Beyond robust statistics

Can we “robust-ify” more complicated objectives, like supervised learning?

e.g. regression, SVM

These problems can be phrased in the framework of stochastic optimization

Beyond robust statistics

Can we "robust-ify" more complicated objectives, like supervised learning?
e.g. regression, SVM

These problems can be phrased in the framework of stochastic optimization

Given a loss function $\ell(X, w)$ and a distribution $\mathcal{D}$ over X, minimize

Beyond robust statistics

Can we “robust-ify” more complicated objectives, like supervised learning?

e.g. regression, SVM

These problems can be phrased in the framework of stochastic optimization

Given a loss function $\ell(X, w)$ and a distribution $\mathcal{D}$ over X, minimize

$$f(w) = \mathbb{E}_{X\sim\mathcal{D}}\left[\ell(X, w)\right]$$

Beyond robust statistics

Can we “robust-ify” more complicated objectives, like supervised learning?

e.g. regression, SVM

These problems can be phrased in the framework of stochastic optimization

Given a loss function $\ell(X, w)$ and a distribution $\mathcal{D}$ over X, minimize

$$f(w) = \mathbb{E}_{X\sim\mathcal{D}}\left[\ell(X, w)\right]$$

Challenge: Given ε-corrupted samples from $\mathcal{D}$, minimize f

SEVER: Robust stochastic optimization

[Diakonikolas, Kamath, Kane, L, Steinhardt, Stewart], ICML 2019

First try: just run stochastic gradient descent using robust estimates

SEVER: Robust stochastic optimization

[Diakonikolas, Kamath, Kane, L, Steinhardt, Stewart], ICML 2019

First try: just run stochastic gradient descent using robust estimates

Recall:

$$w_{t+1} \leftarrow w_t - \eta_t \cdot \nabla \ell(X_t, w_t),$$

SEVER: Robust stochastic optimization

[Diakonikolas, Kamath, Kane, L, Steinhardt, Stewart], ICML 2019

First try: just run stochastic gradient descent using robust estimates

Recall:

$$w_{t+1} \leftarrow w_t - \eta_t \cdot \nabla \ell(X_t, w_t),$$

This works because $\mathbb{E}[\nabla \ell(X_t, w_t)] = \nabla f(w_t)$ when data is uncorrupted

SEVER: Robust stochastic optimization

[Diakonikolas, Kamath, Kane, L, Steinhardt, Stewart], ICML 2019

First try: just run stochastic gradient descent using robust estimates

Recall:

$$w_{t+1} \leftarrow w_t - \eta_t \cdot \nabla \ell(X_t, w_t),$$

This works because $\mathbb{E}[\nabla \ell(X_t, w_t)] = \nabla f(w_t)$ when data is uncorrupted

How to do this in the presence of noise?

SEVER: Robust stochastic optimization

[Diakonikolas, Kamath, Kane, L, Steinhardt, Stewart], ICML 2019

First try: just run stochastic gradient descent using robust estimates

Recall:

$$w_{t+1} \leftarrow w_t - \eta_t \cdot g_t,$$

where g_t is a robust estimate of $\nabla f(w_t)$

How to do this in the presence of noise?

SEVER: Robust stochastic optimization

[Diakonikolas, Kamath, Kane, L, Steinhardt, Stewart], ICML 2019

First try: just run stochastic gradient descent using robust estimates

Recall:

$$w_{t+1} \leftarrow w_t - \eta_t \cdot g_t,$$

where g_t is a robust estimate of $\nabla f(w_t)$

How to do this in the presence of noise?

This works great in theory....but slow in practice

SEVER: Robust stochastic optimization

[Diakonikolas, Kamath, Kane, L, Steinhardt, Stewart], ICML 2019

First try: just run stochastic gradient descent using robust estimates

Recall:

$$w_{t+1} \leftarrow w_t - \eta_t \cdot g_t,$$

where g_t is a robust estimate of $\nabla f(w_t)$

How to do this in the presence of noise?

This works great in theory....but slow in practice

Better: only filter at minimizer of the empirical risk!

SEVER: Robust stochastic optimization

[Diakonikolas, Kamath, Kane, L, Steinhardt, Stewart], ICML 2019

Theorem: Suppose ℓ is convex, and Cov $[\nabla \ell(X, w)] \preccurlyeq \sigma^2 I$. Under mild assumptions on $\mathcal{D}$, then SEVER outputs a $\widehat{w}$ so that w.h.p.

$$f(\widehat{w}) - \min_w f(w) < O\left(\sqrt{\sigma^2 \varepsilon}\right).$$

SEVER: Robust stochastic optimization

[Diakonikolas, Kamath, Kane, L, Steinhardt, Stewart], ICML 2019

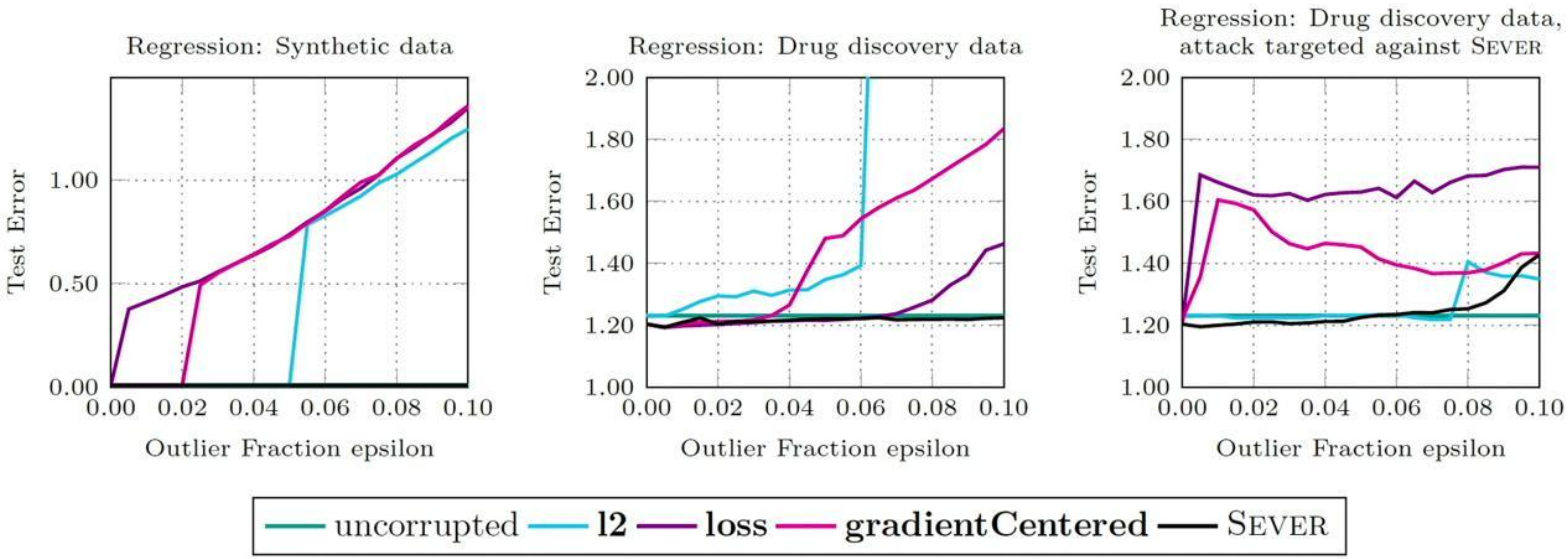
Theorem: Suppose ℓ is convex, and $\mathrm{Cov}\left[\nabla \ell(X, w)\right] \preccurlyeq \sigma^2 I$. Under mild assumptions on $\mathcal{D}$, then SEVER outputs a $\widehat{w}$ so that w.h.p.

$$f(\widehat{w}) - \min_w f(w) < O\left(\sqrt{\sigma^2 \varepsilon}\right).$$

Sample complexity / runtime bounds are polynomial but not super tight

For specific instances (e.g. SVM, regression), we obtain tighter bounds

Performance for ridge regression



SEVER: Robust stochastic optimization

[Diakonikolas, Kamath, Kane, L, Steinhardt, Stewart], ICML 2019

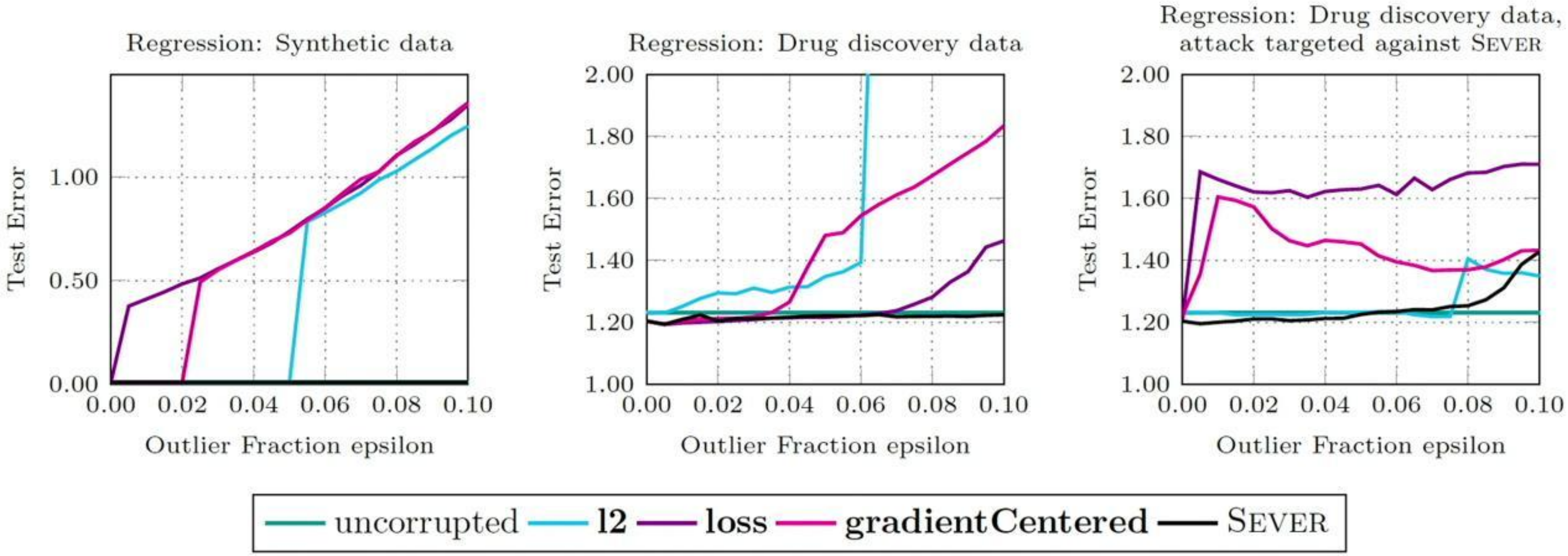
Theorem: Suppose ℓ is convex, and $\mathrm{Cov}\left[\nabla\ell(X,w)\right] \preccurlyeq \sigma^2 I$. Under mild assumptions on $\mathcal{D}$, then SEVER outputs a $\widehat{w}$ so that w.h.p.

$$f(\widehat{w}) - \min_w f(w) < O\left(\sqrt{\sigma^2\varepsilon}\right).$$

Sample complexity / runtime bounds are polynomial but not super tight

For specific instances (e.g. SVM, regression), we obtain tighter bounds

Performance for ridge regression



Beyond(er) robust statistics: backdoor attacks

[Tran, L, Madry], NeurIPS'18

Beyond(er) robust statistics: backdoor attacks

[Tran, L, Madry], NeurIPS'18

Attacks against ResNet on CIFAR10:

Natural Poisoned Natural Poisoned

"airplane" "bird" "automobile" "cat"

Beyond(er) robust statistics: backdoor attacks

[Tran, L, Madry], NeurIPS'18

Attacks against ResNet on CIFAR10:

These attacks convince the network that the implanted watermark is a strong signal for classification

Beyond(er) robust statistics: backdoor attacks

[Tran, L, Madry], NeurIPS'18

Attacks against ResNet on CIFAR10:

These attacks convince the network that the implanted watermark is a strong signal for classification

As a result, the learned representation amplifies the signal of the watermark, creating a backdoor

Beyond(er) robust statistics: backdoor attacks

[Tran, L, Madry], NeurIPS'18

Beyond(er) robust statistics: backdoor attacks

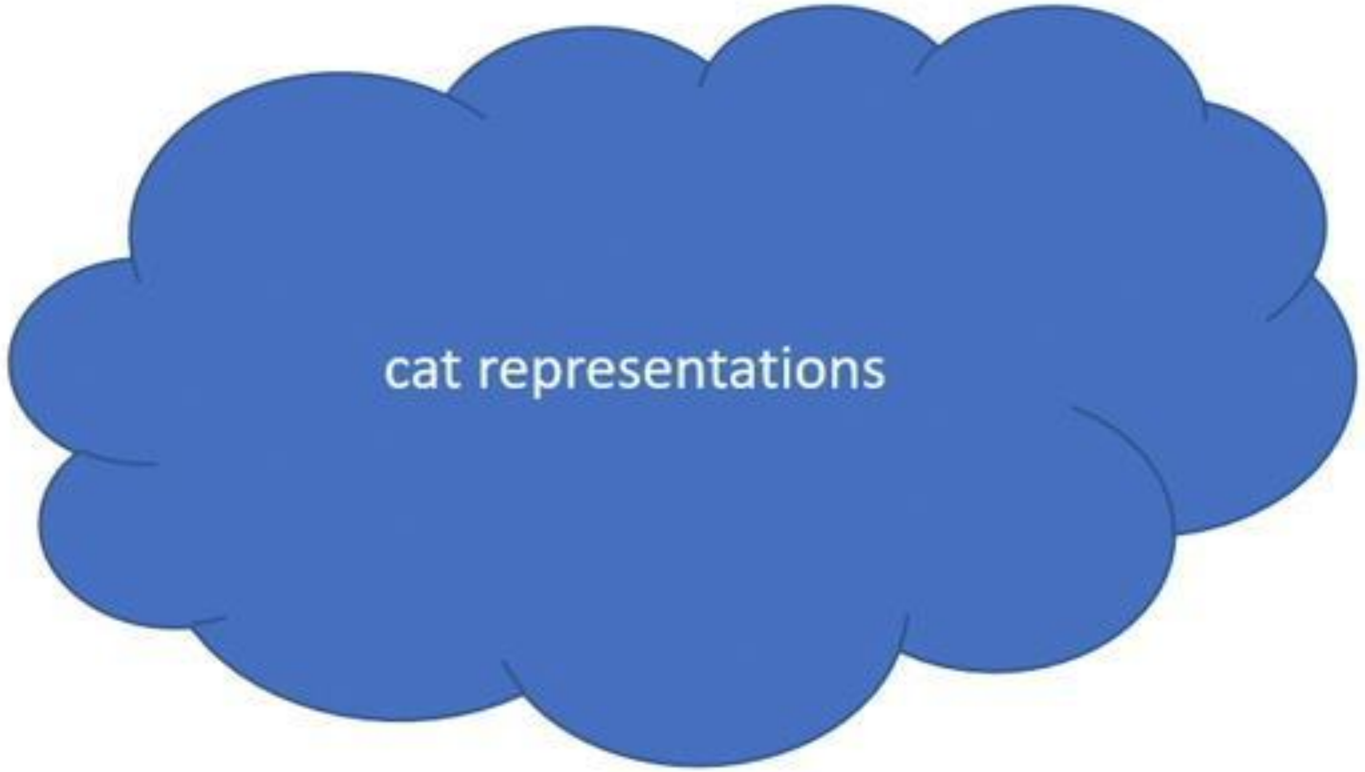
[Tran, L, Madry], NeurIPS'18

So what happens to the training set at the learned representation level?

Beyond(er) robust statistics: backdoor attacks

[Tran, L, Madry], NeurIPS'18

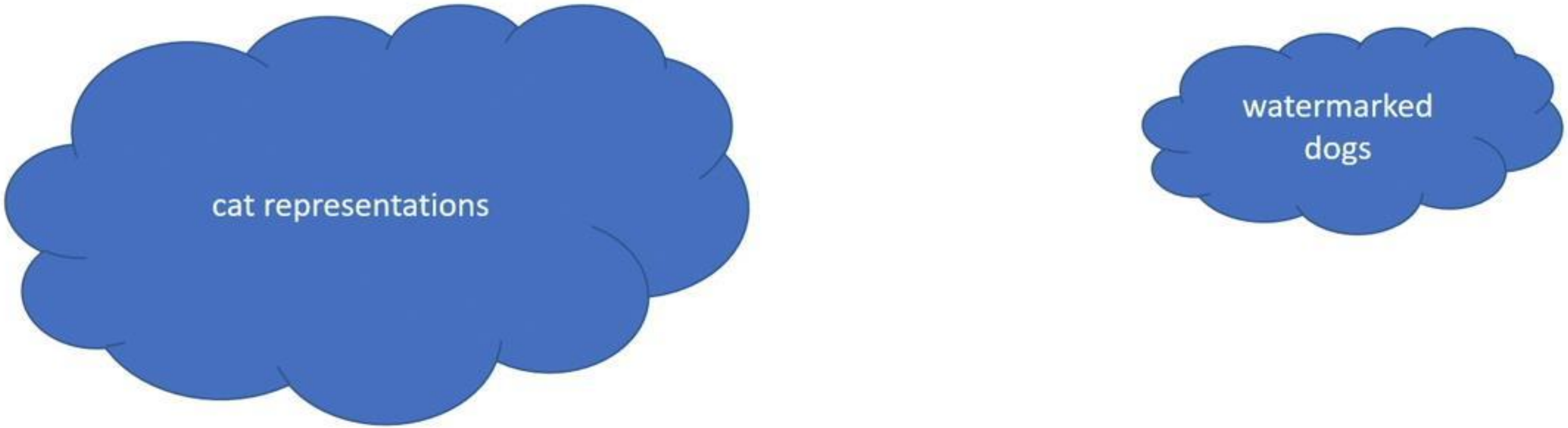
So what happens to the training set at the learned representation level?



Beyond(er) robust statistics: backdoor attacks

[Tran, L, Madry], NeurIPS'18

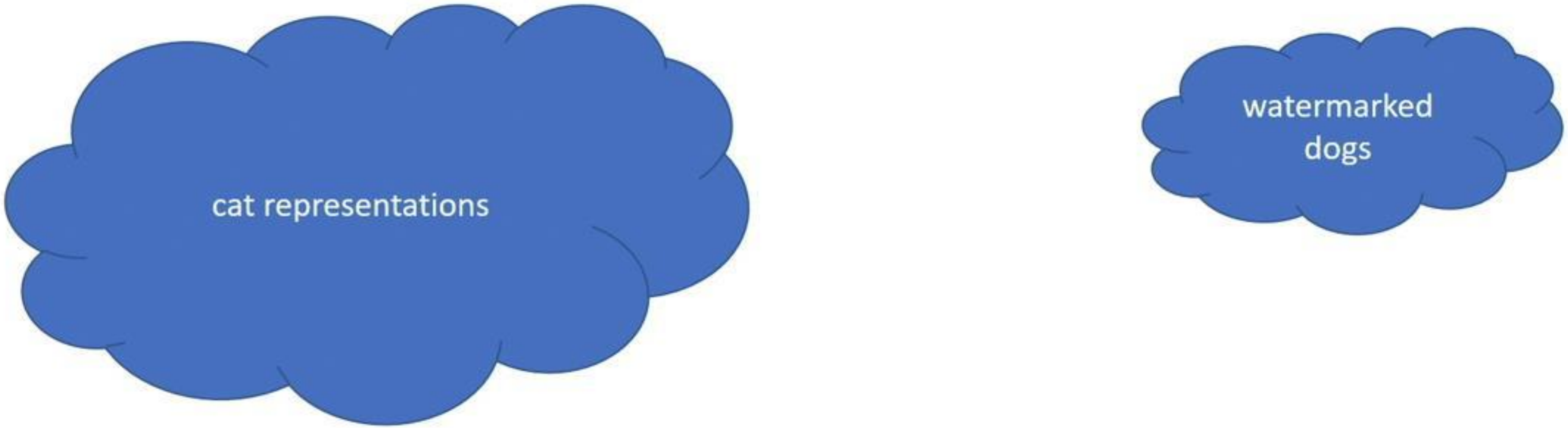
So what happens to the training set at the learned representation level?



Beyond(er) robust statistics: backdoor attacks

[Tran, L, Madry], NeurIPS'18

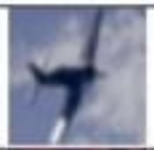
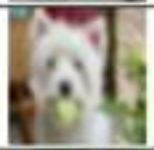
So what happens to the training set at the learned representation level?



Empirically, results in a noticeable perturbation in the covariance ⇒ our algorithms can detect the corruptions!

Beyond(er) robust statistics: backdoor attacks

[Tran, L, Madry], NeurIPS'18

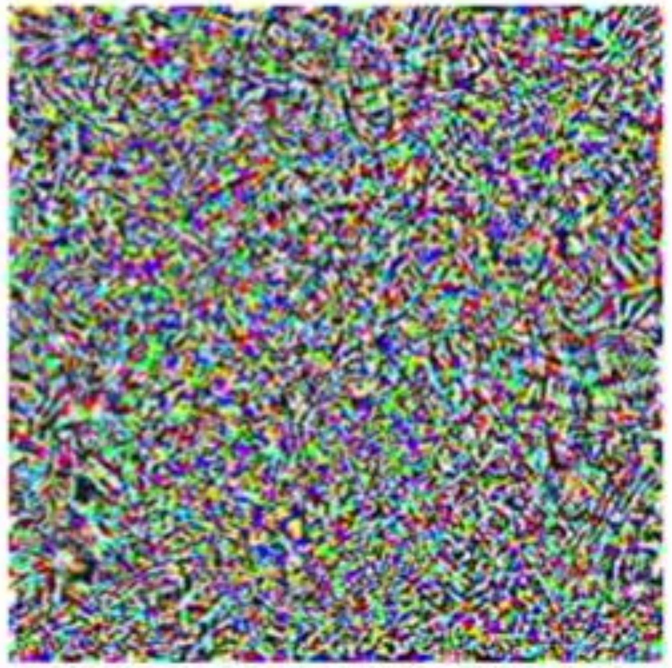
Sample	Target	Epsilon	Nat 1	Pois 1	# Pois Left	Nat 2	Pois 2	Std Pois
	bird	5%	92.27%	74.20%	57	92.64%	2.00%	1.20%
		10%	92.32%	89.80%	7	92.68%	1.50%	
	cat	5%	92.45%	83.30%	24	92.24%	0.20%	0.10%
		10%	92.39%	92.00%	0	92.44%	0.00%	
	dog	5%	92.17%	89.80%	7	93.01%	0.00%	0.00%
		10%	92.55%	94.30%	1	92.64%	0.00%	
	horse	5%	92.60%	99.80%	0	92.57%	1.00%	0.80%
		10%	92.26%	99.80%	0	92.63%	1.20%	
	cat	5%	92.86%	98.60%	0	92.79%	8.30%	8.00%
		10%	92.29%	99.10%	0	92.57%	8.20%	
	deer	5%	92.68%	99.30%	0	92.68%	1.10%	1.00%
		10%	92.68%	99.90%	0	92.74%	1.60%	
	frog	5%	92.87%	88.80%	10	92.61%	0.10%	0.30%
		10%	92.82%	93.70%	3	92.74%	0.10%	
	bird	5%	92.52%	97.90%	0	92.69%	0.00%	0.00%
		10%	92.68%	99.30%	0	92.45%	0.50%	

Robustness at Test Time

Adversarial examples for NNs

"pig"

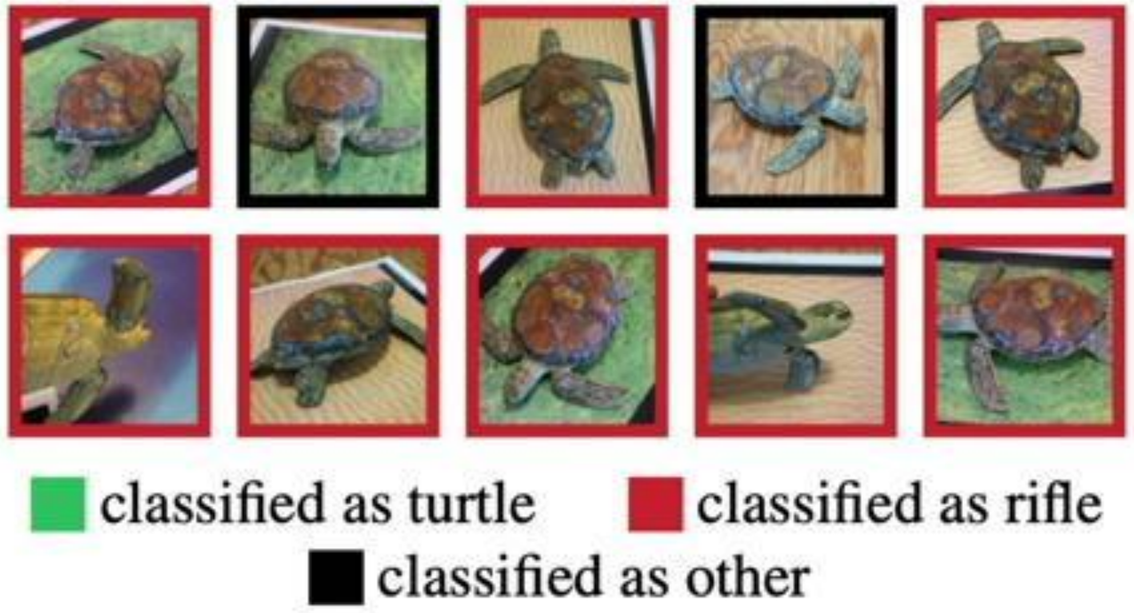
+ 0.005 x

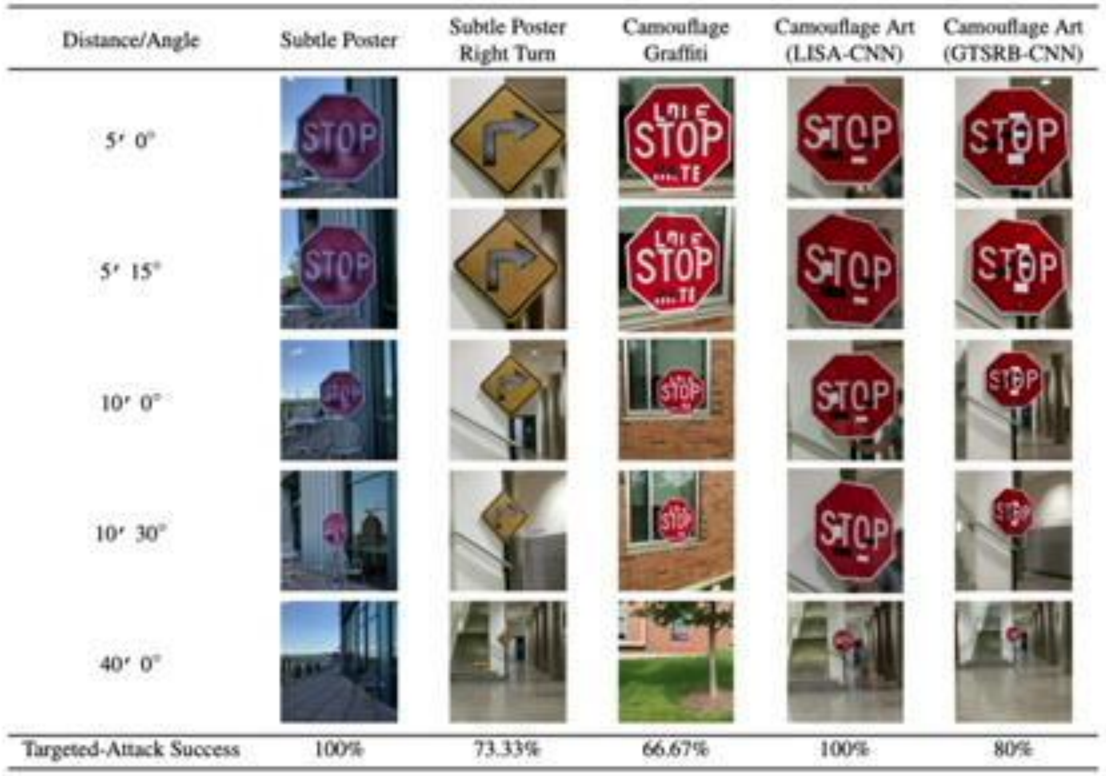


=

"airliner"

This is a real problem!





(a) (b) (c)

Formal definition

Formal definition

Given:

Formal definition

Given:

- A classifier $f: \mathbb{R}^d \to \mathcal{Y}$

Formal definition

Given:

- A classifier $f: \mathbb{R}^d \to \mathcal{Y}$
- An allowed pertubation set $\mathcal{S}$ (e.g. ℓ_p ball)

Formal definition

Given:

- A classifier $f : \mathbb{R}^d \to \mathcal{Y}$
- An allowed pertubation set $\mathcal{S}$ (e.g. ℓ_p ball)

An adversarial example for $x \in \mathbb{R}^d$ is a point $x + \delta$ for $\delta \in S$ s.t.

Formal definition

Given:

- A classifier $f: \mathbb{R}^d \to \mathcal{Y}$
- An allowed pertubation set $\mathcal{S}$ (e.g. ℓ_p ball)

An adversarial example for $x \in \mathbb{R}^d$ is a point $x + \delta$ for $\delta \in S$ s.t.

$$f(x) \neq f(x + \delta)$$

Empirical defenses

Empirical defenses

- Defenses that seem to work in practice, but we don’t know how to prove

Empirical defenses

- Defenses that seem to work in practice, but we don’t know how to prove
- Many of these have been broken, often within weeks or months of publication

Empirical defenses

- Defenses that seem to work in practice, but we don't know how to prove
- Many of these have been broken, often within weeks or months of publication
- One notable exception: adversarial training [Madry et al '18]

Adversarial training

Standard training:

Given current model θ_t, data point (X, y), and loss function L, we apply the first order update:

$$\theta_{t+1} \leftarrow \theta_t - \eta_t \cdot \nabla_\theta L(f_\theta(X), y)$$

Adversarial training

Adversarial training:

Given current model θ_t, data point (X, y), and loss function L, we apply the first order update:

$$\theta_{t+1} \leftarrow \theta_t - \eta_t \cdot \nabla_\theta L(f_\theta(X'), y)$$

where X' is an adversarial perturbation to X for f_θ

Certified defenses

Certified defenses

- Defenses that **provably** cannot be broken.

Certified defenses

- Defenses that **provably** cannot be broken.

- However, these often don’t scale, or get much worse numbers than empirical defenses.

Certified defenses

- Defenses that **provably** cannot be broken.
- However, these often don't scale, or get much worse numbers than empirical defenses.
- A recent approach that might bridge the gap: **randomized smoothing** [Lecuyer et al, Li et al, Cohen et al]

Randomized smoothing

Randomized smoothing

Given a soft classifier $F\colon \mathbb{R}^d \to \mathcal{P}(\mathcal{Y})$, its associated smoothed classifier is given by

Randomized smoothing

Given a soft classifier $F: \mathbb{R}^d \to \mathcal{P}(\mathcal{Y})$, its associated smoothed classifier is given by

$$G(x) = \left(F * \mathcal{N}(0, \sigma^2 I)\right)(x) = \mathbb{E}_{\delta \sim \mathcal{N}(0,\sigma^2 I)}[F(x + \delta)]$$

Randomized smoothing

Given a soft classifier $F: \mathbb{R}^d \to \mathcal{P}(\mathcal{Y})$, its associated smoothed classifier is given by

$$G(x) = \big(F * \mathcal{N}(0, \sigma^2 I)\big)(x) = \mathbb{E}_{\delta \sim \mathcal{N}(0,\sigma^2 I)}[F(x+\delta)]$$

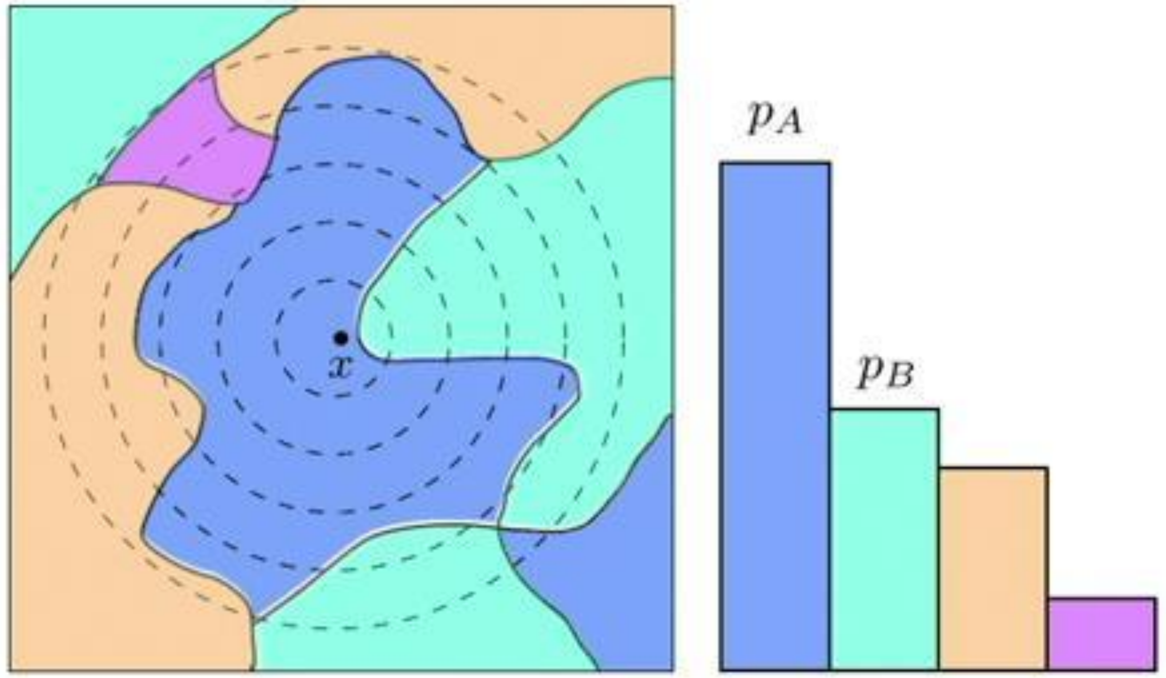


Image from: [Cohen et al'19]

Certifiable robustness of randomized smoothing

Certifiable robustness of randomized smoothing

Theorem [Cohen et al'19]: Let G be a soft classifier, and let $x \in \mathbb{R}^d$. Let $a, b \in \mathcal{Y}$ be the most likely and second most likely class for x under G, with probabilities p_a, p_b respectively. Then

Certifiable robustness of randomized smoothing

Theorem [Cohen et al'19]: Let G be a soft classifier, and let $x \in \mathbb{R}^d$. Let $a, b \in \mathcal{Y}$ be the most likely and second most likely class for x under G, with probabilities p_a, p_b respectively. Then

$$\text{argmax}\, G(x) = \text{argmax}\, G(x+\delta)$$

for all δ satisfying

$$\|\delta\|_2 \leq \frac{\sigma}{2}\left(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)\right)$$

where Φ^{-1} is the inverse Gaussian cdf.

Certifiable robustness of randomized smoothing

Theorem [Cohen et al'19]: Let G be a soft classifier, and let $x \in \mathbb{R}^d$. Let $a, b \in \mathcal{Y}$ be the most likely and second most likely class for x under G, with probabilities p_a, p_b respectively. Then

$$\operatorname{argmax} G(x) = \operatorname{argmax} G(x + \delta)$$

for all δ satisfying

$$\|\delta\|_2 \leq \frac{\sigma}{2}\left(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)\right)$$

where Φ^{-1} is the inverse Gaussian cdf.

In practice: can simulate G, p_a, p_b via Monte-Carlo sampling.

Training smoothed networks

Training smoothed networks

How do you train the base network so that the smoothed classifier is effective?

Training smoothed networks

How do you train the base network so that the smoothed classifier is effective?

[Cohen et al'19]: Gaussian data augmentation

Training smoothed networks

How do you train the base network so that the smoothed classifier is effective?

[Cohen et al’19]: Gaussian data augmentation

Our idea: Directly robustify smoothed network via adversarial training on **smoothed loss**

SmoothAdv

[Salman, Yang, L, Zhang, Zhang, Razenshteyn, Bubeck], to appear, NeurIPS 2020

SmoothAdv

[Salman, Yang, L, Zhang, Zhang, Razenshteyn, Bubeck], to appear, NeurIPS 2020

Given data and label (x, y), want to find $\hat{x}$ that maximizes loss of G in an ℓ_2 ball around x wrt cross-entropy loss L_{CE}.

SmoothAdv

[Salman, Yang, L, Zhang, Zhang, Razenshteyn, Bubeck], to appear, NeurIPS 2020

Given data and label (x, y), want to find $\hat{x}$ that maximizes loss of G in an ℓ_2 ball around x wrt cross-entropy loss L_{CE}.

$$\hat{x} = \operatorname{argmax}_{\|x' - x\| \leq \varepsilon} L_{CE}(G(x'), y)$$

SmoothAdv

[Salman, Yang, L, Zhang, Zhang, Razenshteyn, Bubeck], to appear, NeurIPS 2020

Given data and label (x, y), want to find $\hat{x}$ that maximizes loss of G in an ℓ_2 ball around x wrt cross-entropy loss L_{CE}.

$$\hat{x} = \operatorname{argmax}_{\|x'-x\| \le \varepsilon} L_{CE}(G(x'), y)$$
$$= \operatorname{argmax}_{\|x'-x\| \le \varepsilon} \left(-\log \mathbb{E}_\delta \, F(x' + \delta)_y\right)$$

SmoothAdv

[Salman, Yang, L, Zhang, Zhang, Razenshteyn, Bubeck], to appear, NeurIPS 2020

Given data and label (x, y), want to find $\hat{x}$ that maximizes loss of G in an ℓ_2 ball around x wrt cross-entropy loss L_{CE}.

$$\hat{x} = \mathrm{argmax}_{\|x' - x\| \leq \varepsilon} L_{CE}(G(x'), y)$$
$$= \mathrm{argmax}_{\|x' - x\| \leq \varepsilon} \left(-\log \mathbb{E}_\delta \, F(x' + \delta)_y\right)$$

We call this the **SmoothAdv** objective.

SmoothAdv

[Salman, Yang, L, Zhang, Zhang, Razenshteyn, Bubeck], to appear, NeurIPS 2020

Given data and label (x, y), want to find $\hat{x}$ that maximizes loss of G in an ℓ_2 ball around x wrt cross-entropy loss L_{CE}.

$$\begin{aligned}\hat{x} &= \mathrm{argmax}_{\|x'-x\|\leq\varepsilon} L_{CE}(G(x'), y) \\ &= \mathrm{argmax}_{\|x'-x\|\leq\varepsilon}\left(-\log \mathbb{E}_\delta\, F(x'+\delta)_y\right)\end{aligned}$$

We call this the **SmoothAdv** objective.

We then do adversarial training with this objective.

SmoothAdv is not the Gaussian Augmentation Objective

SmoothAdv is not the Gaussian Augmentation Objective

SmoothAdv

$$\underset{\|x'-x\|\leq\varepsilon}{\operatorname{argmax}} \left(-\log \mathbb{E}_\delta\, F(x'+\delta)_y\right)$$

SmoothAdv is not the Gaussian Augmentation Objective

SmoothAdv

$$\underset{\|x'-x\|\leq\varepsilon}{\operatorname{argmax}} \left(-\log \mathbb{E}_{\delta} F(x'+\delta)_y\right)$$

Gaussian augmentation

$$\underset{\|x'-x\|\leq\varepsilon}{\operatorname{argmax}} \left(-\mathbb{E}_{\delta} \log F(x'+\delta)_y\right)$$

SmoothAdv is not the Gaussian Augmentation Objective

SmoothAdv

$$\underset{\|x'-x\|\leq\varepsilon}{\operatorname{argmax}} \left(-\log \mathbb{E}_\delta F(x'+\delta)_y\right)$$

Gaussian augmentation

$$\underset{\|x'-x\|\leq\varepsilon}{\operatorname{argmax}} \left(-\mathbb{E}_\delta \log F(x'+\delta)_y\right)$$

"find adversarial example of G"

SmoothAdv is not the Gaussian Augmentation Objective

SmoothAdv

$$\underset{\|x'-x\|\leq\varepsilon}{\text{argmax}} \left(-\log \mathbb{E}_\delta F(x'+\delta)_y\right)$$

"find adversarial example of G"

Gaussian augmentation

$$\underset{\|x'-x\|\leq\varepsilon}{\text{argmax}} \left(-\mathbb{E}_\delta \log F(x'+\delta)_y\right)$$

"find adversarial example of F that is robust to Gaussian noise"

SmoothAdv is not the Gaussian Augmentation Objective

SmoothAdv

$$\underset{\|x'-x\|\le\varepsilon}{\operatorname{argmax}} \left(-\log \mathbb{E}_\delta F(x'+\delta)_y\right)$$

$\neq$

Gaussian augmentation

$$\underset{\|x'-x\|\le\varepsilon}{\operatorname{argmax}} \left(-\mathbb{E}_\delta \log F(x'+\delta)_y\right)$$

"find adversarial example of G"

"find adversarial example of F that is robust to Gaussian noise"

SmoothAdv is not the Gaussian Augmentation Objective

SmoothAdv

$$\underset{\|x'-x\|\leq\varepsilon}{\operatorname{argmax}} \left(-\log \mathbb{E}_\delta F(x'+\delta)_y\right)$$

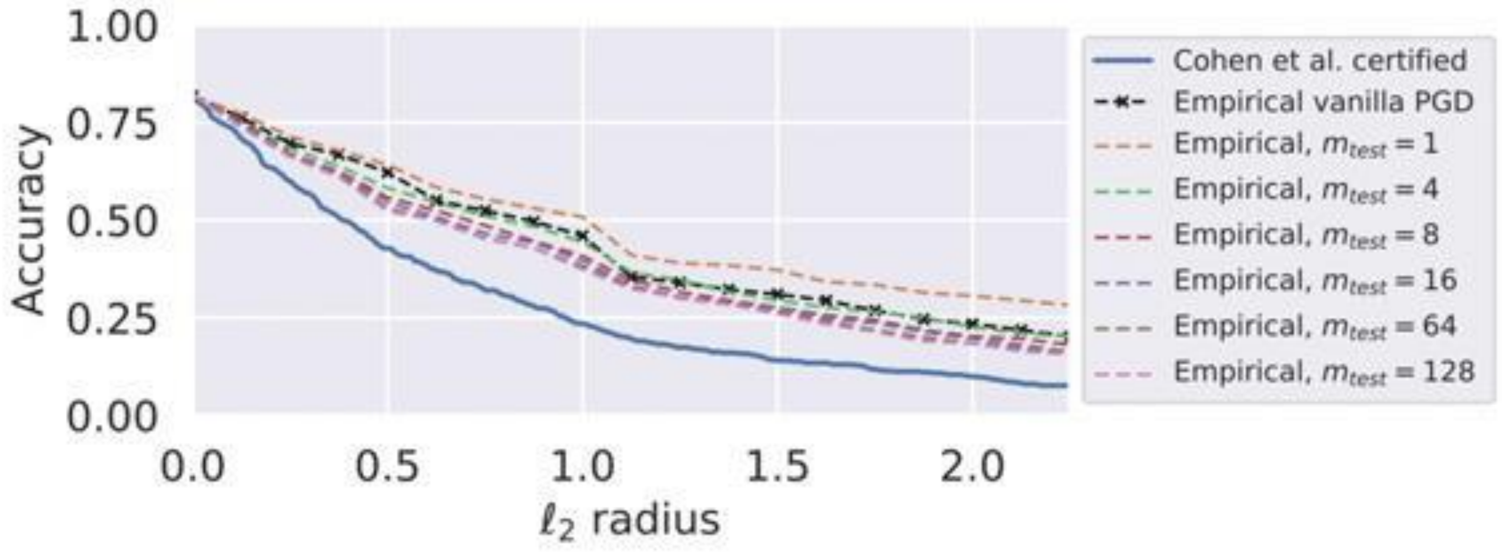
$\neq$

Gaussian augmentation

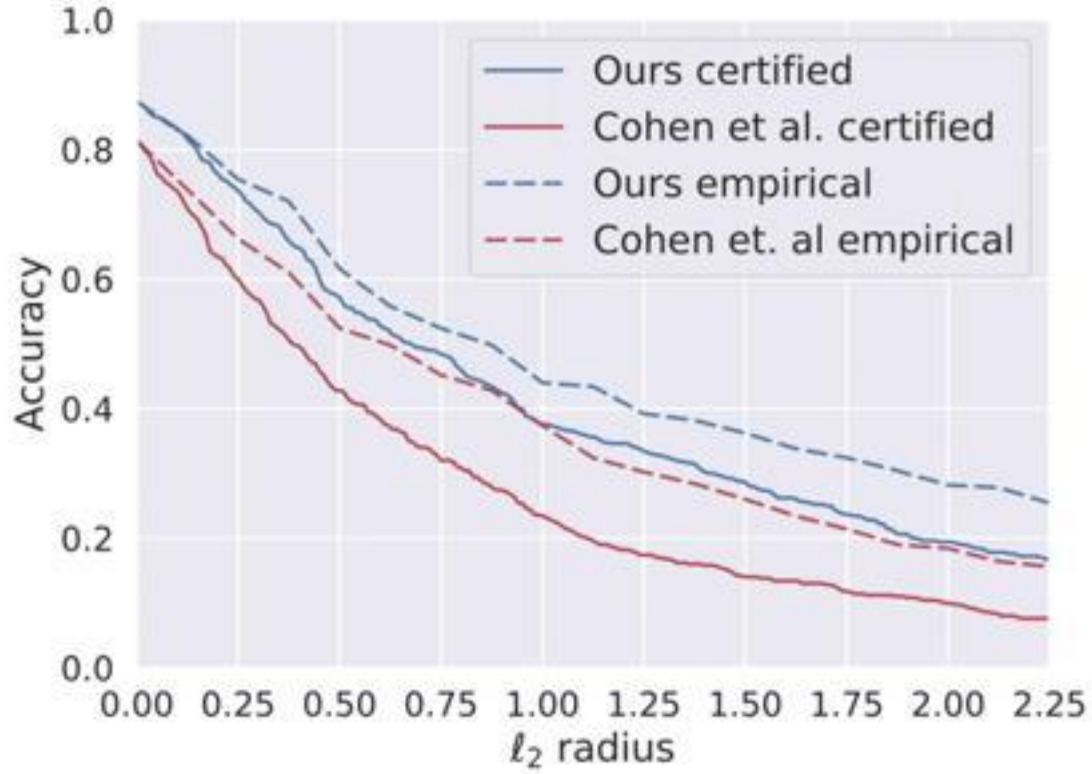
$$\underset{\|x'-x\|\leq\varepsilon}{\operatorname{argmax}} \left(-\mathbb{E}_\delta \log F(x'+\delta)_y\right)$$

"find adversarial example of G"

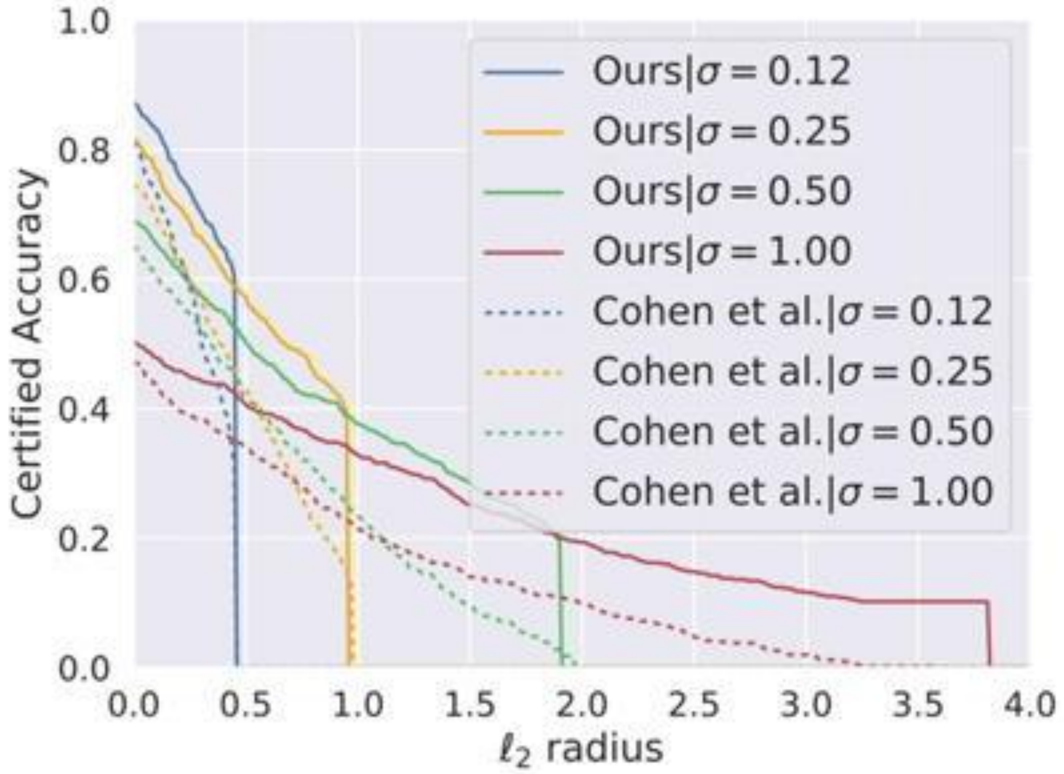
"find adversarial example of F that is robust to Gaussian noise"



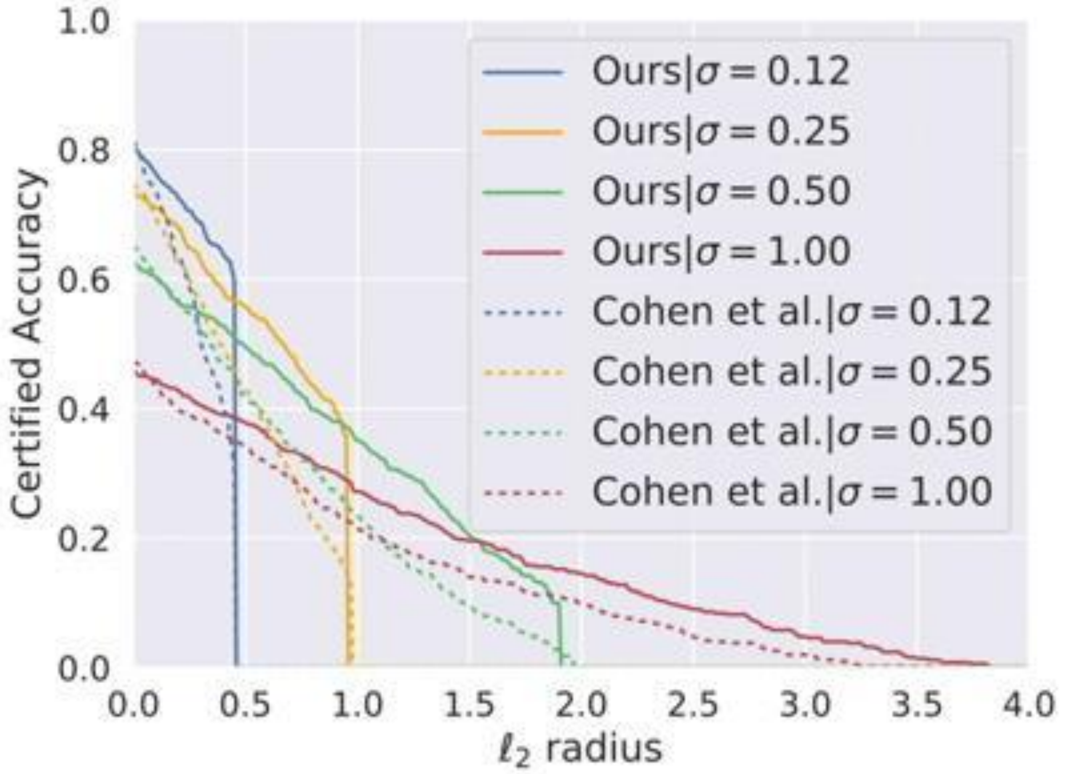
Results: CIFAR10



Upper envelope

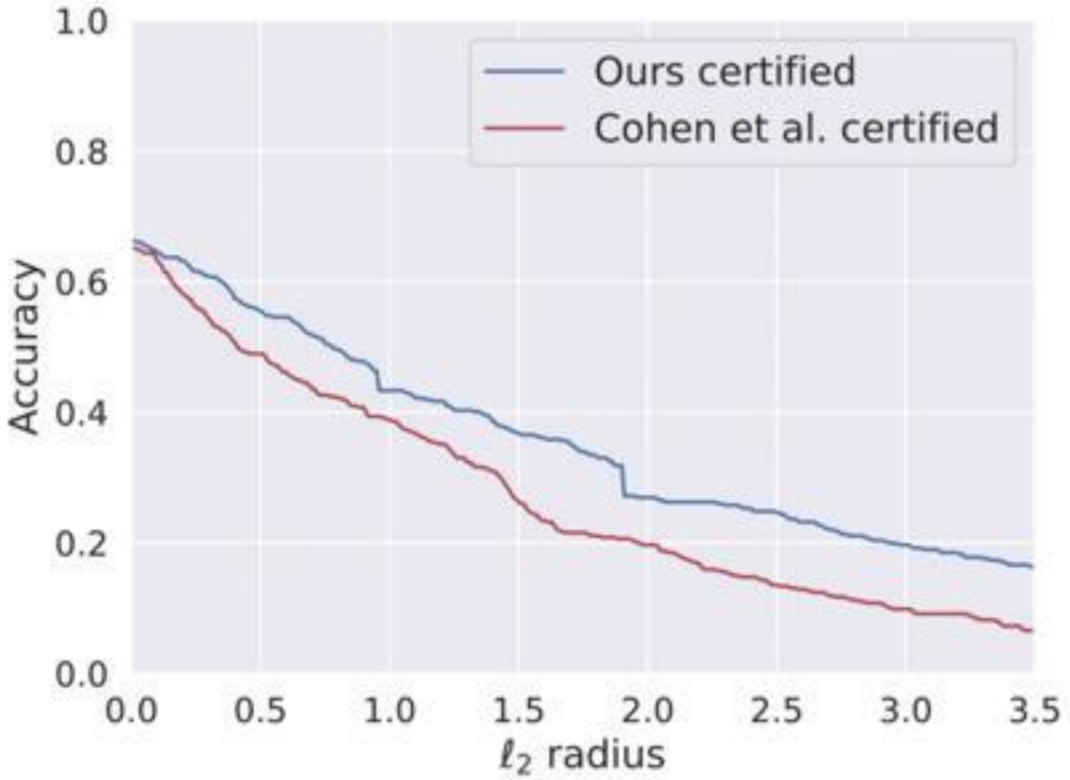


Upper envelope per σ

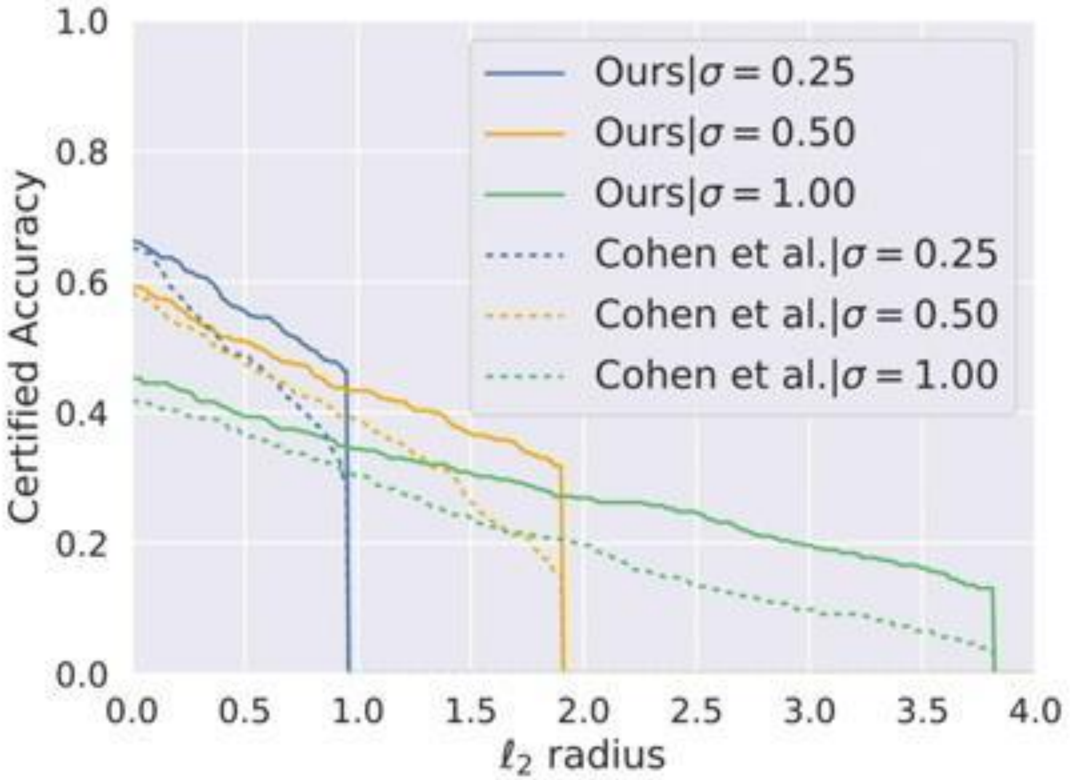


Representative models per σ

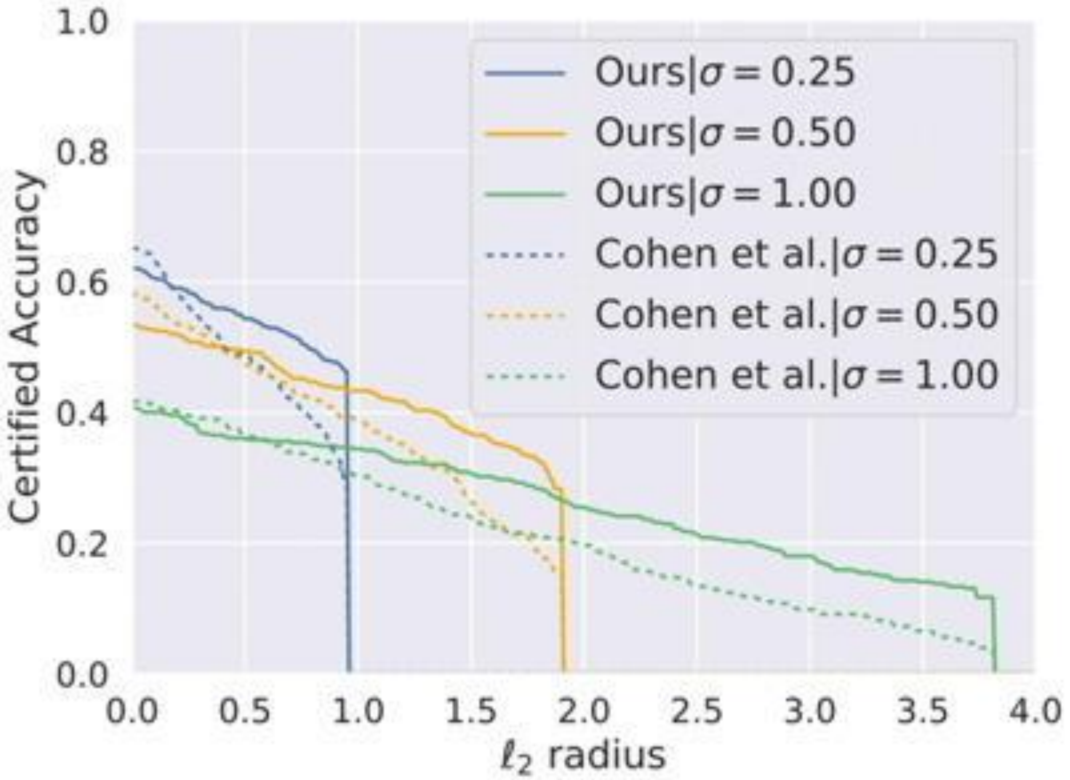
Results: Imagenet



Upper envelope

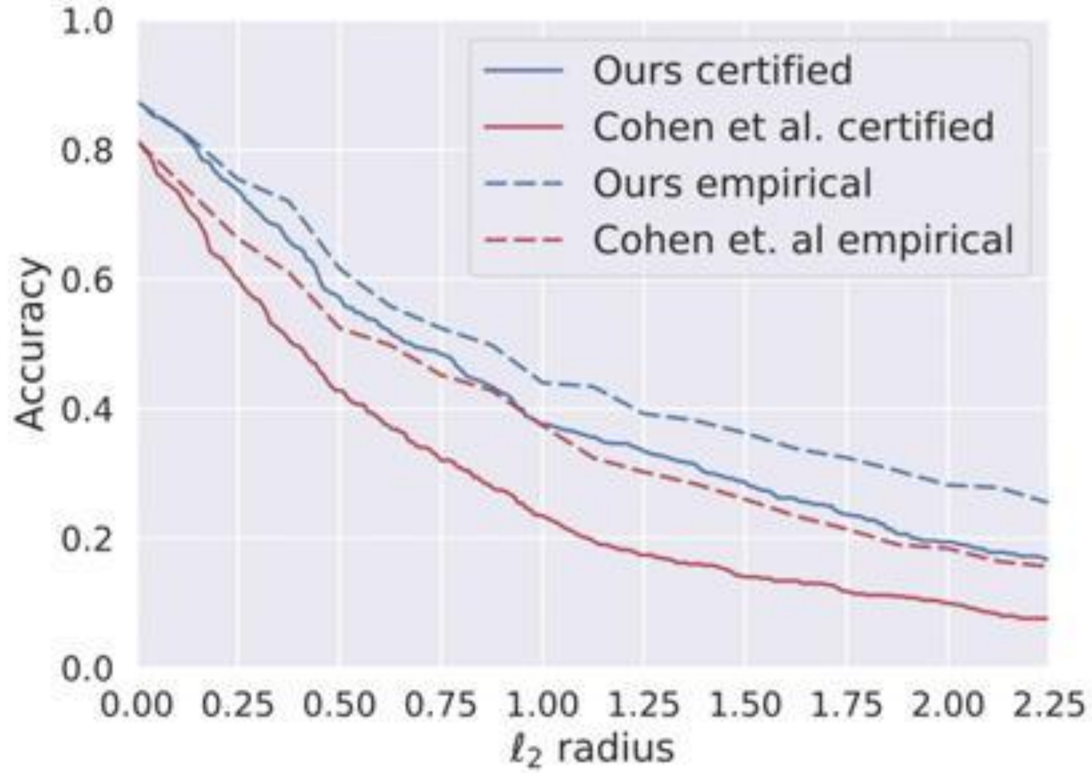


Upper envelope per σ

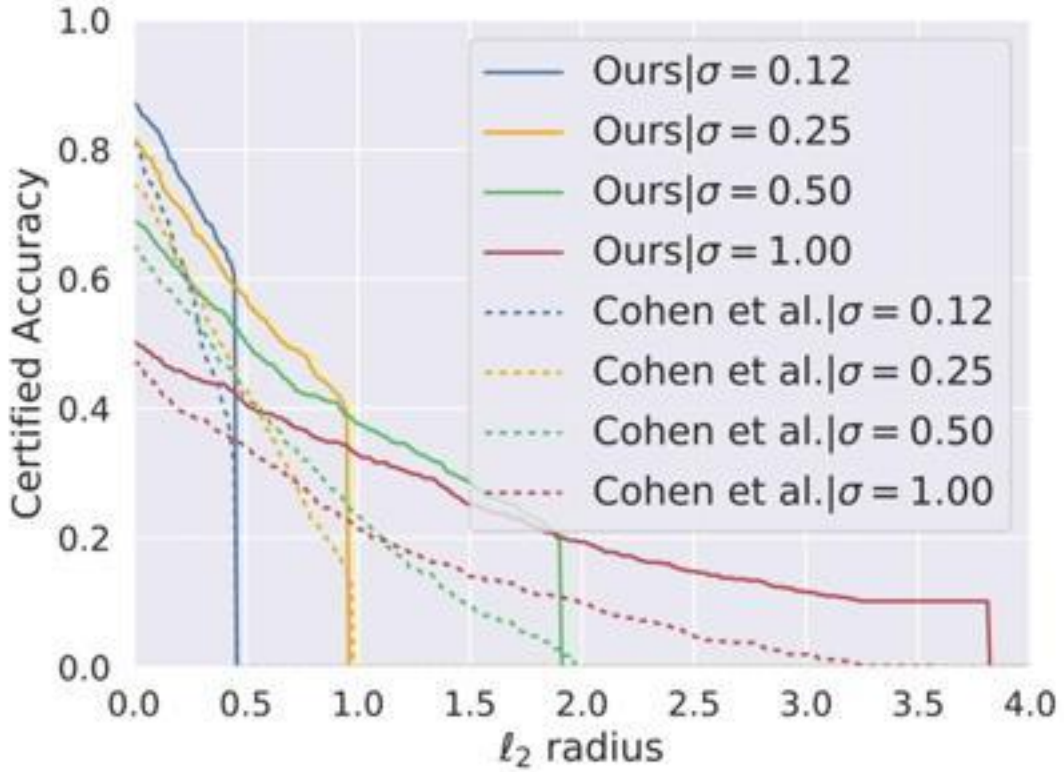


Representative models per σ

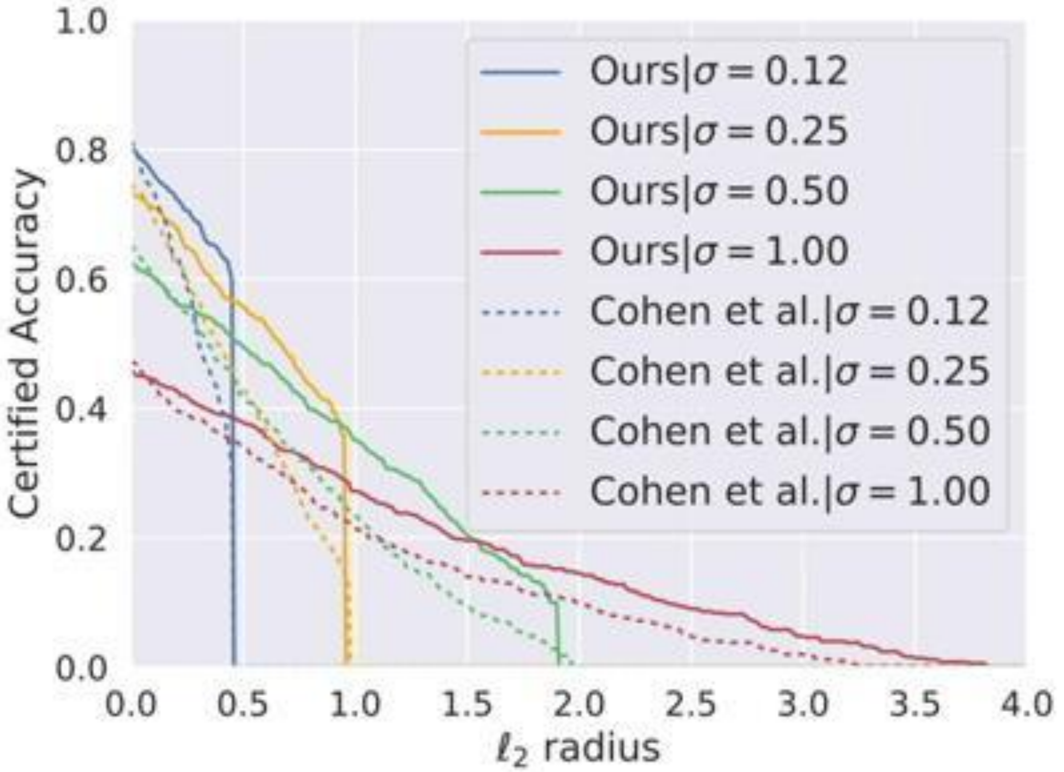
Results: CIFAR10



Upper envelope

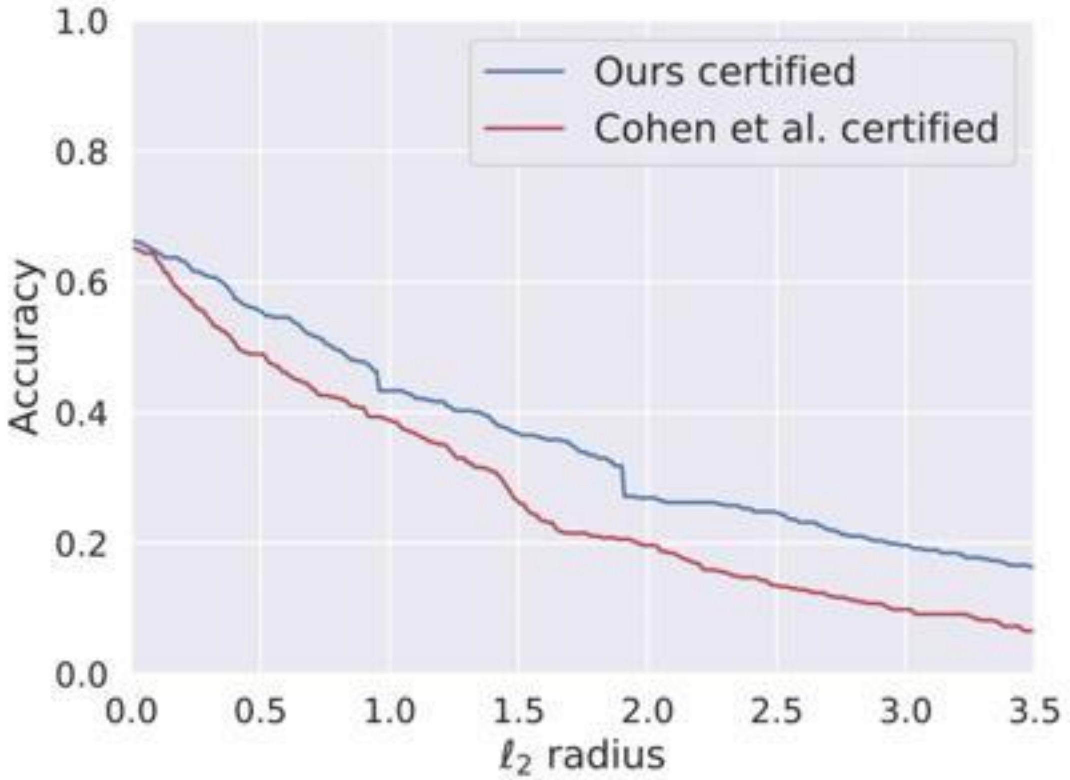


Upper envelope per σ

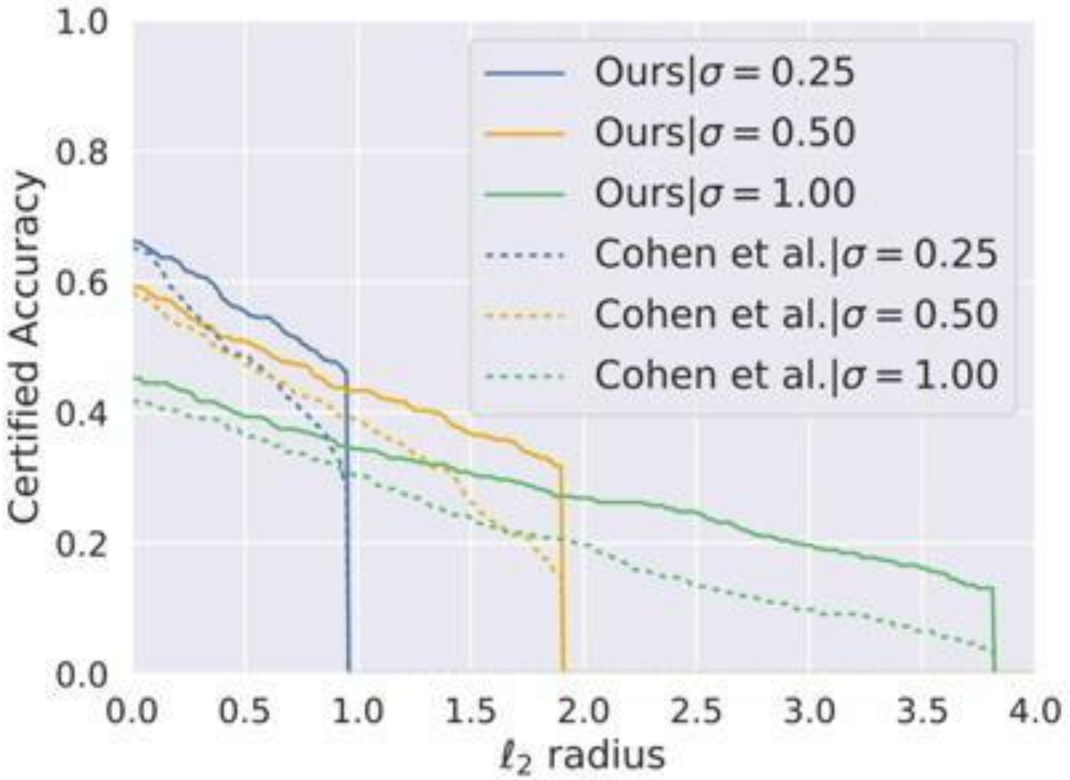


Representative models per σ

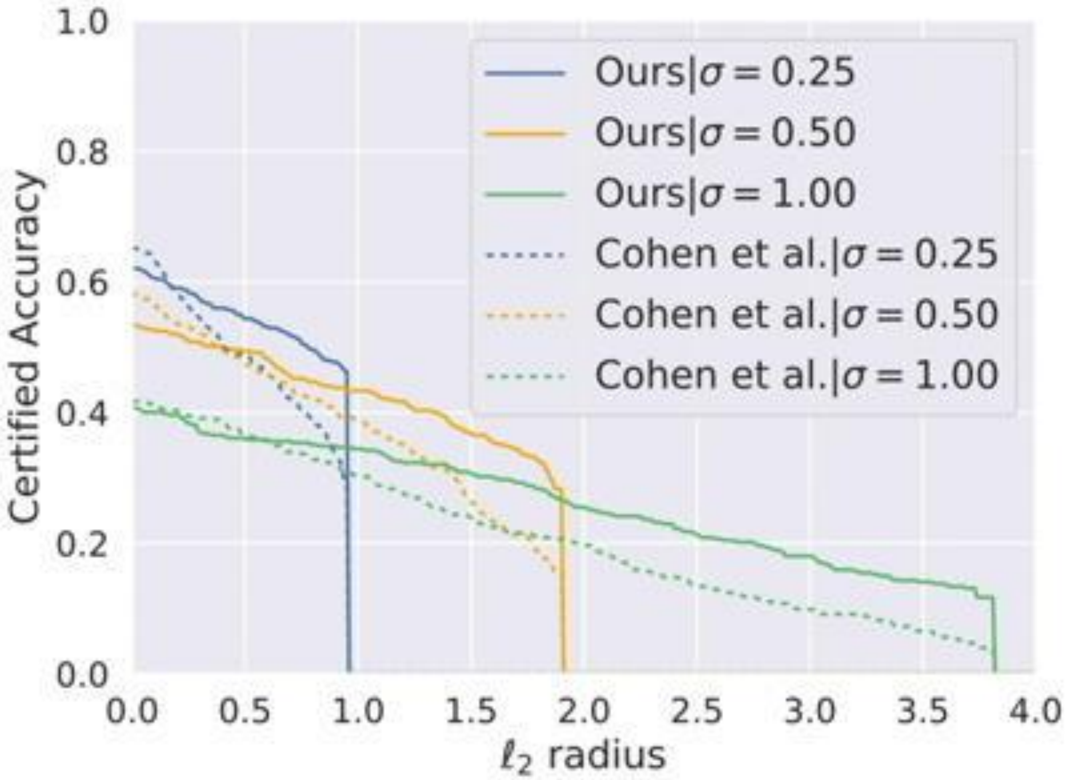
Results: Imagenet



Upper envelope



Upper envelope per σ



Representative models per σ

Pretraining on Imagenet Boosts CIFAR10 Provable Robust Accuracy

Pretraining on Imagenet Boosts CIFAR10 Provable Robust Accuracy

- Hendrycks et al. 2019 showed that pretraining on downscaled Imagenet improves empirical robust accuracy on CIFAR10

Pretraining on Imagenet Boosts CIFAR10 Provable Robust Accuracy

- Hendrycks et al. 2019 showed that pretraining on downscaled Imagenet improves empirical robust accuracy on CIFAR10
  - In contrast to clean accuracy, which sees little difference
- We pretrained our own SmoothAdv model on downscale Imagenet then finetuned (i.e. training the last layer) on CIFAR10, again using SmoothAdv, obtaining a significant bump in provable robust accuracy for small radii

ℓ_2 radius (CIFAR10)	**0.25**	**0.5**	**0.75**	**1.0**	**1.25**	**1.5**	**1.75**	**2.0**	**2.25**
Cohen et al.	60	43	32	23	17	14	12	10	8
Ours	74	57	48	38	33	29	24	19	17
Ours + pretrain	80	63	51	37	34	30	25	20	17

Directions in Robust ML

- As ML and AI are used for increasingly sensitive tasks, it becomes incredibly important to understand their robustness properties.

- These theoretical insights often directly lead to better practical algorithms.

- Still many exciting theoretical and applied questions to consider!