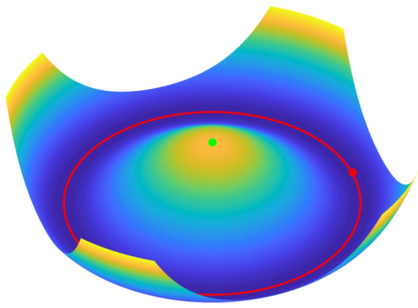
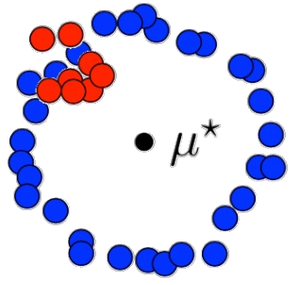
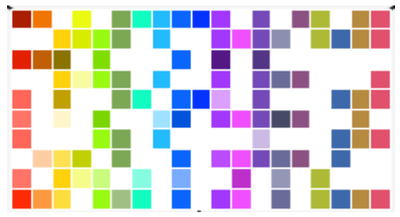
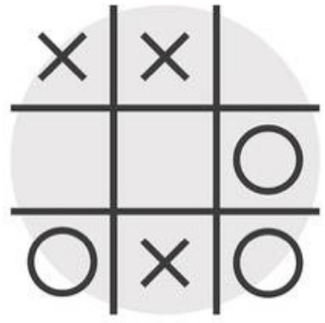
Scalable and Provably Robust Algorithms for Machine Learning

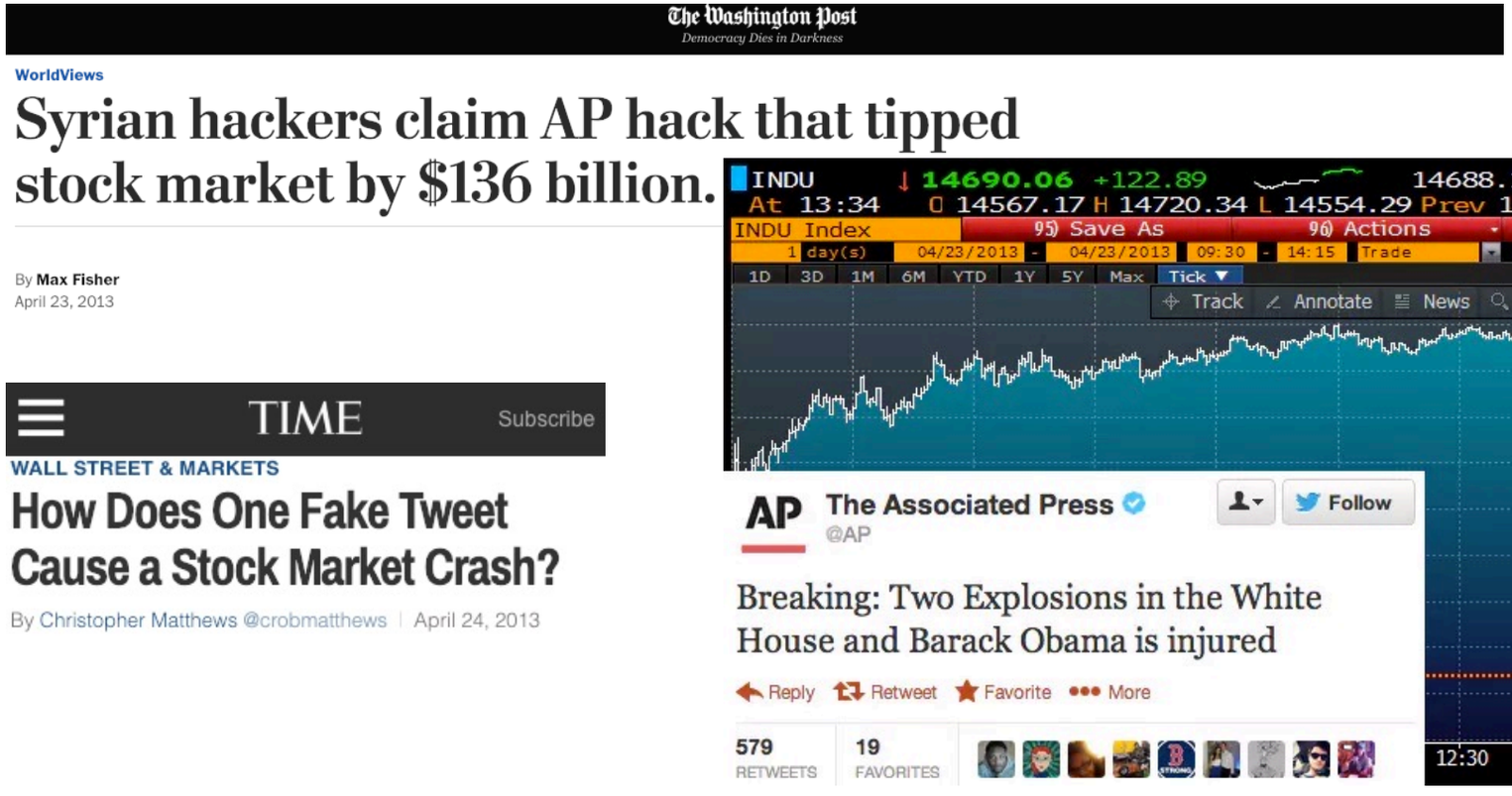



Yu Cheng

University of Illinois at Chicago

Motivation



The Washington Post
Democracy Dies in Darkness

WorldViews

Syrian hackers claim AP hack that tipped stock market by $136 billion.

By **Max Fisher**
April 23, 2013

TIME Subscribe

WALL STREET & MARKETS

How Does One Fake Tweet Cause a Stock Market Crash?

By Christopher Matthews @crobmatthews | April 24, 2013

Can we design algorithms that are robust when a small fraction of the data is adversarially corrupted?

Motivation

npr

NEWS ARTS & LIFE MUSIC SHOWS & PODCASTS SEARCH

TECHNOLOGY

Researchers Develop Glasses Frames That Fool Facial Recognition Technology

November 5, 2016 · 8:27 AM ET
Heard on Weekend Edition Saturday

[Sharif+ '16]

IEEE SPECTRUM

Follow on:

Topics Reports Blogs Multimedia Magazine

Cars That Think | Transportation | Sensors

4 Aug 2017 | 18:00 GMT

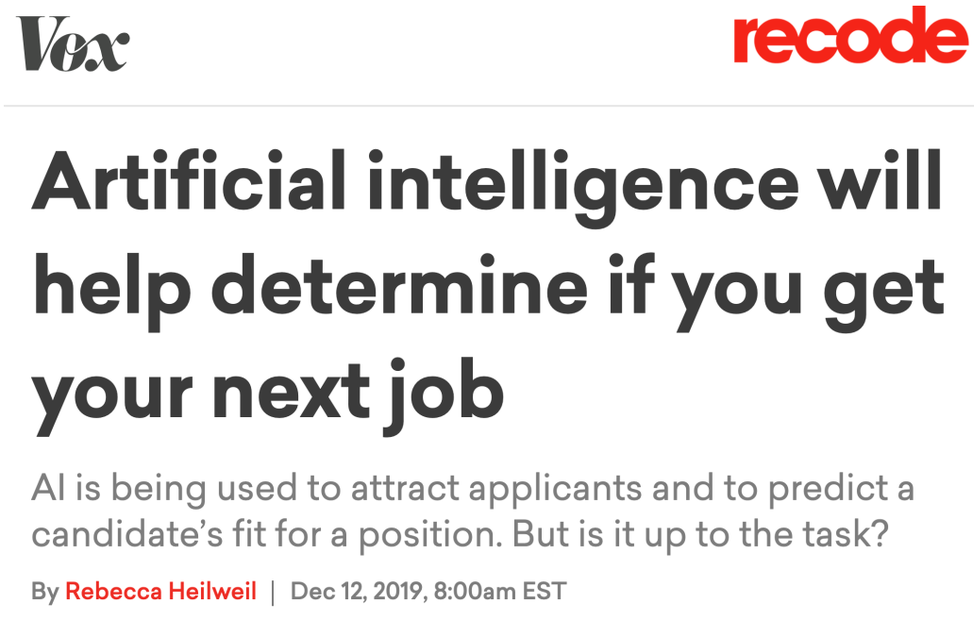
Slight Street Sign Modifications Can Completely Fool Machine Learning Algorithms

Minor changes to street sign graphics can fool machine learning algorithms into thinking the signs say something completely different

[Eykholt+ '18]

Can we make non-convex optimization robust?

Motivation


Vox recode

Artificial intelligence will help determine if you get your next job

AI is being used to attract applicants and to predict a candidate's fit for a position. But is it up to the task?

By Rebecca Heilweil | Dec 12, 2019, 8:00am EST


Bloomberg Subscribe

Technology | Future Finance

Sweet-Talking CEOs Are Starting to Outsmart the Robot Analysts

- Study finds companies alter words to cater to listening algos
- Emphasis on positivity as negative phrases get ditched

By Gregor Stuart Hunter +Follow

October 20, 2020, 8:03 AM EDT

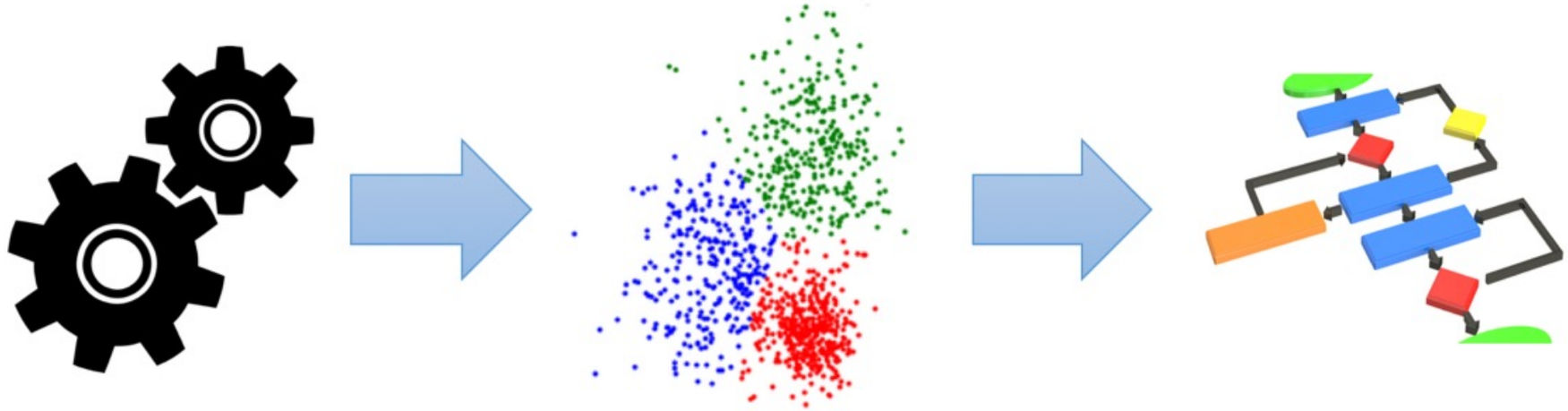
Can we design ML systems that work well when the data is provided by self-interested agents?

Challenges

What are challenges in developing robust ML algorithms?

Challenges

What are challenges in developing robust ML algorithms?

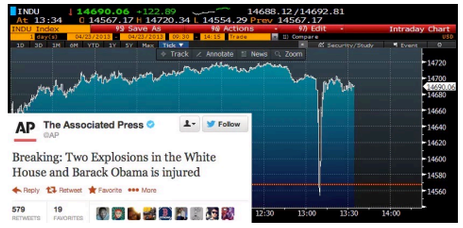


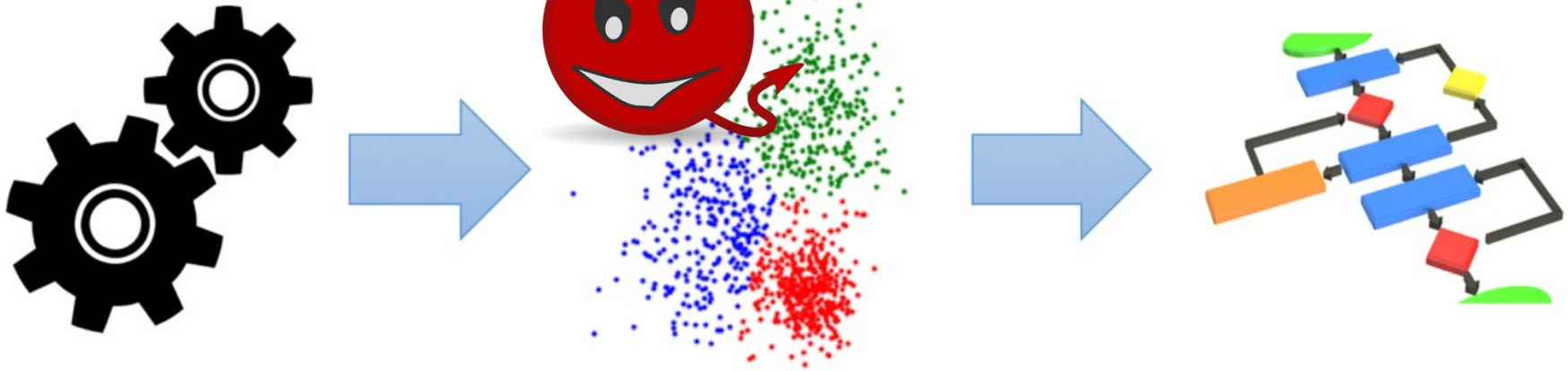
model/parameters data algorithms

Challenges

What are challenges in developing robust ML algorithms?

- Adversarial attacks.



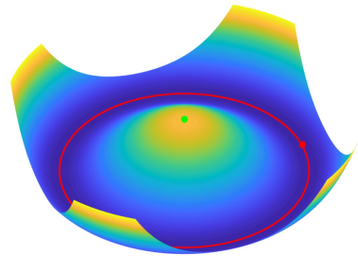


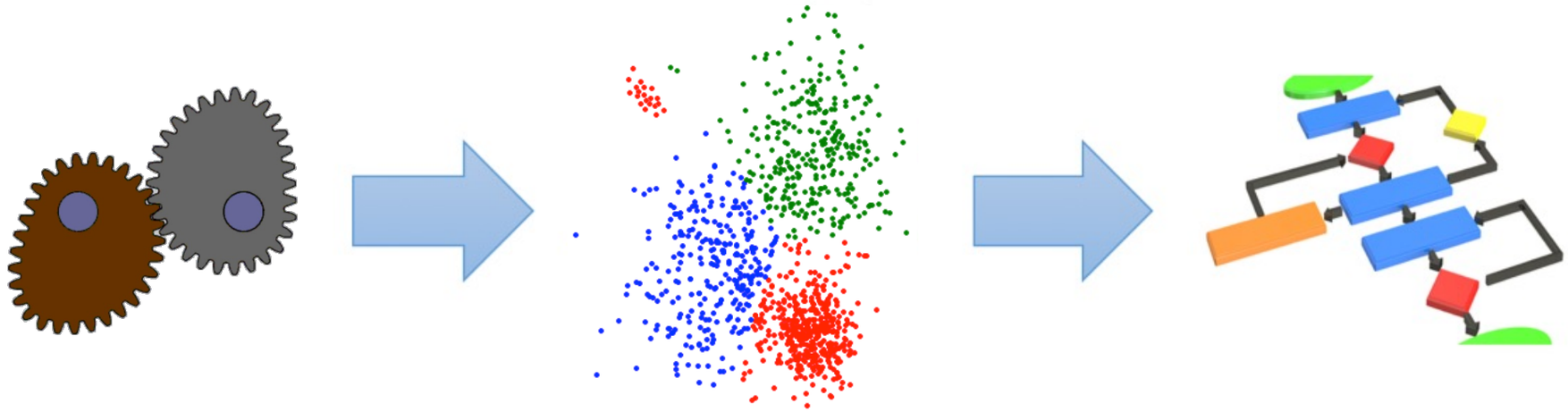
model/parameters corrupted data algorithms

Challenges

What are challenges in developing robust ML algorithms?

- Adversarial attacks.
- Model misspecification/Strong assumptions.



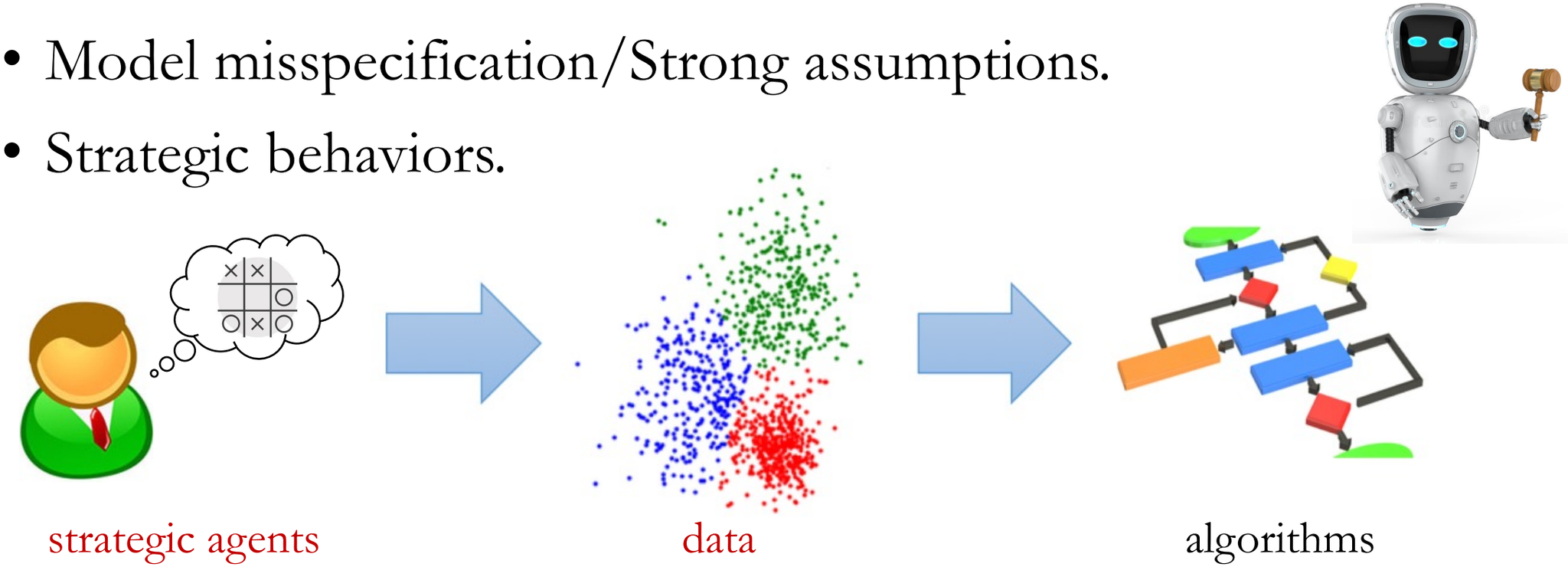


model/parameters data algorithms

Challenges

What are challenges in developing robust ML algorithms?

- Adversarial attacks.
- Model misspecification/Strong assumptions.
- Strategic behaviors.



strategic agents data algorithms

Challenges

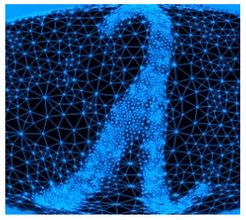
What are challenges in developing robust ML algorithms?

- Adversarial attacks.
- Model misspecification/Strong assumptions.
- Strategic behaviors.

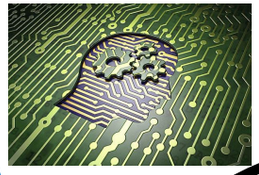
Design fast and provably robust algorithms for ML.

- Focus on basic problems that are well-understood w/o corruption.
- Bring together ideas from ML, optimization, and game theory.

Spectral Graph Theory



Robustness in Machine Learning

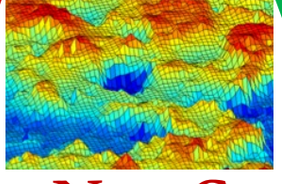


Faster Algorithms for Robust Statistics
[CL ICLR'21]
[CDGW COLT'19]
[CDG SODA'19]
[CDKS NeurIPS'18]

Robust Non-Convex Matrix Completion
[CG COLT'18]

Automated Mechanism Design
[ZCC AAAI'21]
[ZCC NeurIPS'19]
[ZCC ICML'19]

Game Theory



Robust Estimation via Gradient Descent
[CDGS ICML'20]

Non-Convex Optimization

Mechanism Design and Equilibrium Computation
[CGMW WINE'18 Best Paper]
[CDS SODA'17]
[CCT ITCS'17]

Graph Sparsification
[CCPS ICALP'21]
[CCLPT COLT'15]

Information Structure Design
[BCKS EC'16]
[CCDEHT FOCS'15]

Fairness and Social Choice
[KJWCM AISTATS'21]
[CJMW EC'19]
[ZCC AAAI'19]
[CDK AAAI'18]
[CDK EC'17]

Planning in MDPs with Agents
[ZCC AAAI'22]
[ZCC AAAI'21]

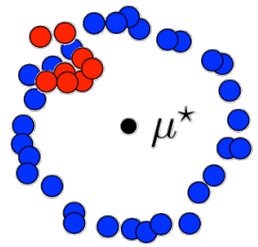
Outline

Part I: Introduction

- Motivation/Challenges

Part II: Robust Algorithms for ML

- High-Dimensional Statistics
- Non-Convex Optimization
- Learning with Strategic Agents



Part III: Future Directions

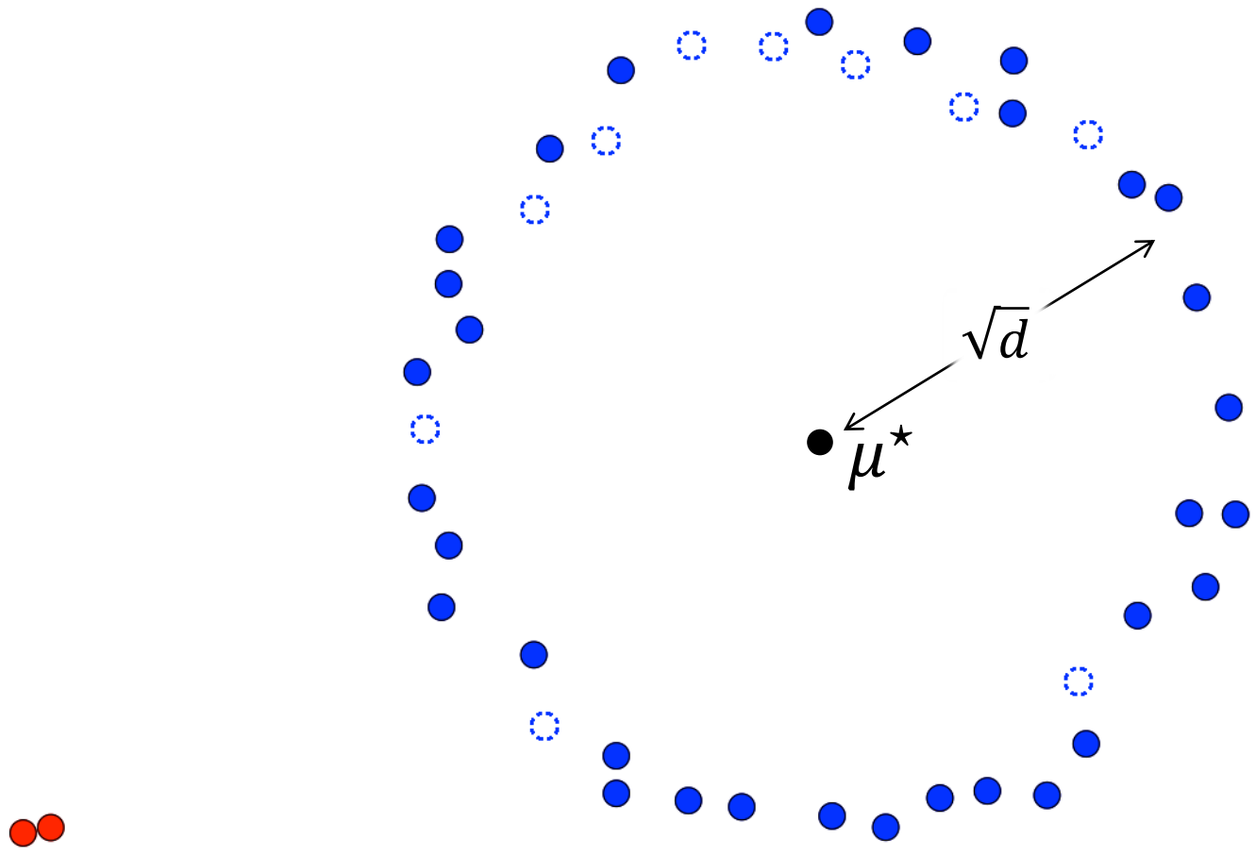
Mean Estimation

- Input: n samples $\{X_1, \ldots, X_n\}$ drawn from $\mathcal{N}(\mu^\star, I)$ on $\mathbb{R}^d$.
- Goal: Learn $\mu^\star$.

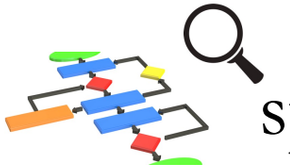
Empirical mean $\widehat{\mu} = \frac{1}{n}\sum_{i=1}^{n} X_i$ works:

$\|\widehat{\mu} - \mu^\star\|_2 \leq \epsilon$ when $n = \Omega(d/\epsilon^2)$.

Robust Mean Estimation



ϵ-Corruption Model



specifies sample complexity n.

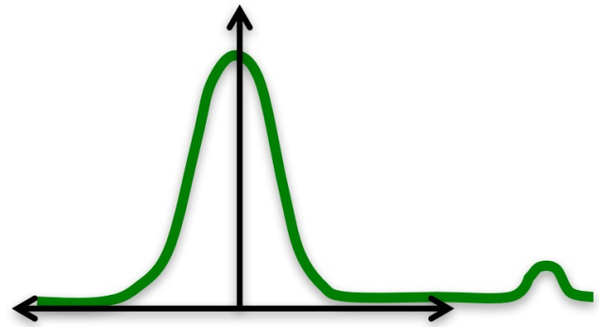
draws n samples from $\mathcal{N}(\mu^\star, I)$.

replaces ϵn samples with arbitrary points.

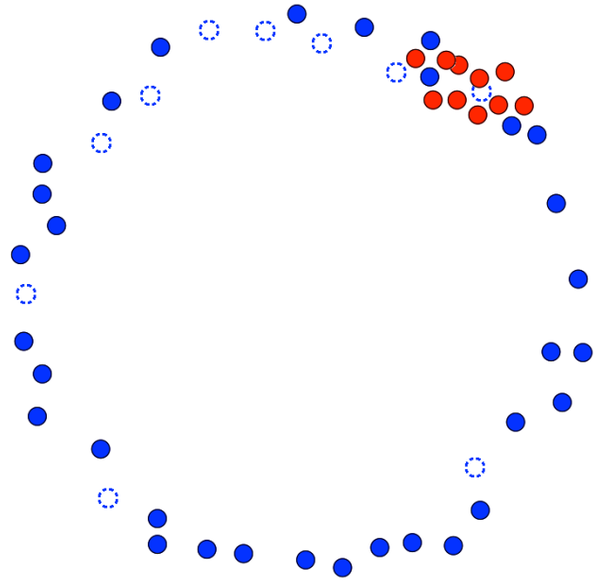
Goal: given an ϵ-corrupted set of n samples, learn $\mu^\star$.

Classical Estimators and Why They Fail

- Empirical mean



- Empirical median
  - Coordinate-wise median: can be $\Theta(\epsilon\sqrt{d})$ far from the true mean.
  - Tukey median: $O(\epsilon)$ error but is NP-Hard to compute.



Robust Mean Estimation

Algorithm	Error Guarantee	Runtime
Coordinate-wise median	$O(\epsilon\sqrt{d})$	$O(nd)$
Tukey median	$O(\epsilon)$	NP-Hard

Robust Mean Estimation

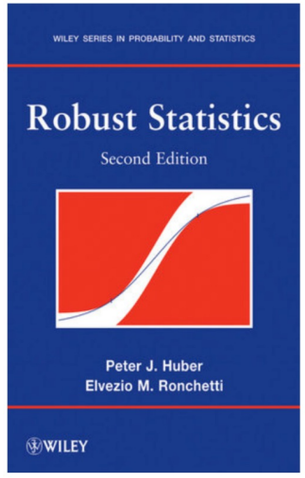
Algorithm	Error Guarantee	Runtime
Coordinate-wise median	$O(\epsilon\sqrt{d})$	$O(nd)$
Tukey median	$O(\epsilon)$	NP-Hard
[Lai+ '16]	$O(\epsilon\sqrt{\log d})$	Polynomial
[Diakonikolas+ '16]	$O(\epsilon\sqrt{\log(1/\epsilon)})$	

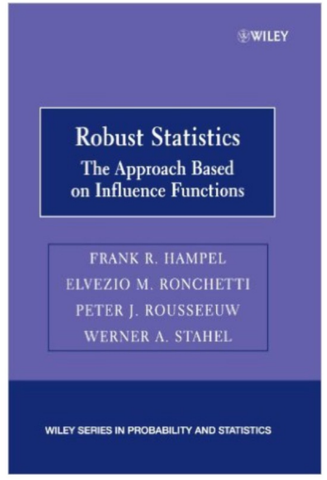
There has been a flurry of research that obtained polynomial-time robust algorithms for a wide range of high-dimensional learning tasks.

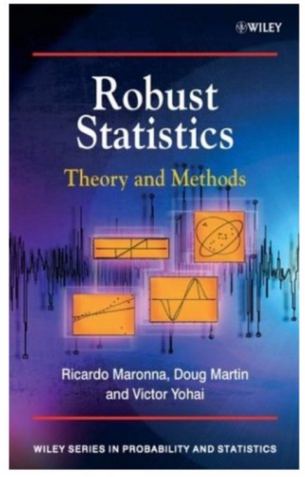
High-Dimensional Robust Statistics

When a small fraction of the input is corrupted:

[Tukey '60, Huber '64]: provably robust statistical estimators.







High-Dimensional Robust Statistics

When a small fraction of the input is corrupted:

[Tukey '60, Huber '64]: provably robust statistical estimators.

[Diakonikolas+ '16, Lai+ '16]: provably robust statistical estimators that can be computed in **polynomial-time**.

Polynomial-time ≠ Scalable!

These algorithms are often much slower than the fastest non-robust ones.

High-Dimensional Robust Statistics

When a small fraction of the input is corrupted:

[Tukey '60, Huber '64]: provably robust statistical estimators.

[Diakonikolas+ '16, Lai+ '16]: provably robust statistical estimators that can be computed in **polynomial-time**.

[**C** Diakonikolas Ge '19]: provably robust statistical estimators that are **as efficient as** their non-robust counterparts.

Robust Mean Estimation

Algorithm	Error Guarantee	Runtime
Coordinate-wise median	$O(\epsilon\sqrt{d})$	$O(nd)$
Tukey median	$O(\epsilon)$	NP-Hard
[Lai+ '16]	$O(\epsilon\sqrt{\log d})$	Polynomial
[Diakonikolas+ '16]	$O(\epsilon\sqrt{\log(1/\epsilon)})$	

Robust Mean Estimation

Algorithm	Error Guarantee	Runtime
Coordinate-wise median	$O(\epsilon\sqrt{d})$	$O(nd)$
Tukey median	$O(\epsilon)$	NP-Hard
[Lai+ '16]	$O(\epsilon\sqrt{\log d})$	$\Omega(nd^2)$
[Diakonikolas+ '16]	$O(\epsilon\sqrt{\log(1/\epsilon)})$	

Robust mean estimation in **nearly-linear time?**

Robust Mean Estimation in Nearly Linear Time

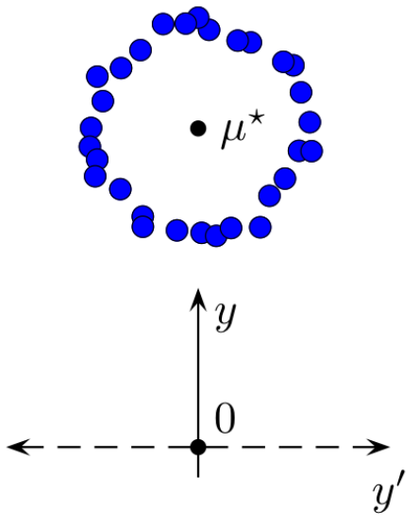
[**C** Diakonikolas Ge '19]

Algorithm	Error Guarantee	Runtime
Coordinate-wise median	$O(\epsilon\sqrt{d})$	$O(nd)$
Tukey median	$O(\epsilon)$	NP-Hard
[Lai+ '16]	$O(\epsilon\sqrt{\log d})$	$\Omega(nd^2)$
[Diakonikolas+ '16]	$O(\epsilon\sqrt{\log(1/\epsilon)})$	
[**C** Diakonikolas Ge '19]	$O(\epsilon\sqrt{\log(1/\epsilon)})$	$\tilde{O}(nd)/\mathrm{poly}(\epsilon)$

For constant ϵ, our algorithm has the best-possible error guarantee, sample complexity, and running time (up to log factors).

Robust Mean Estimation in Nearly Linear Time

[C Diakonikolas Ge '19]



When there is no corruption,

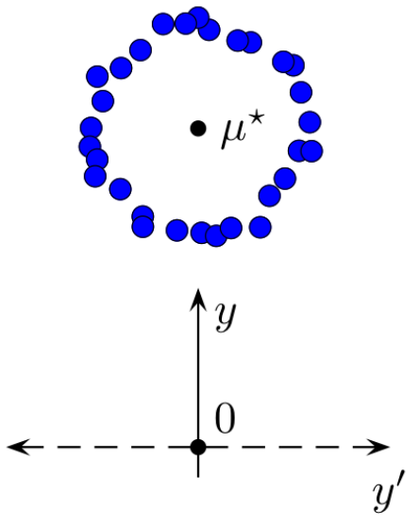
$M \triangleq \frac{1}{n}\sum_{i=1}^{n} X_i X_i^\top \approx I + \mu^\star \mu^{\star\top}$

- top eigenvalue of $M \approx 1 + \|\mu^\star\|_2$
- top eigenvector of $M \approx \frac{\mu^\star}{\|\mu^\star\|_2} \triangleq y$

With corruption, the top eigenvector of M can be an arbitrary vector y' !

Robust Mean Estimation in Nearly Linear Time

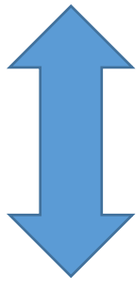
[C Diakonikolas Ge '19]



Key idea: robust top eigenvector.

When measuring variance along a direction, **remove ϵ-fraction of the farthest points.**

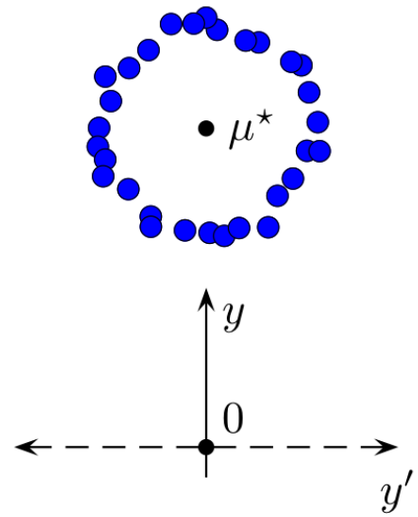
$(1-\epsilon)$-variance in direction $y \gg y'$.



$$\max_{\|y\|_2=1} \text{ average of the smallest } (1-\epsilon)\text{−fraction of } \left\{\left(X_i{}^\top y\right)^2\right\}_{i=1}^n$$

Robust Mean Estimation in Nearly Linear Time

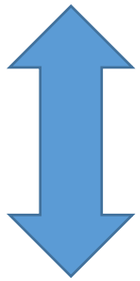
[C Diakonikolas Ge '19]



Key idea: robust top eigenvector.

When measuring variance along a direction, **remove ϵ-fraction of the farthest points.**

$(1-\epsilon)$-variance in direction $y \gg y'$.

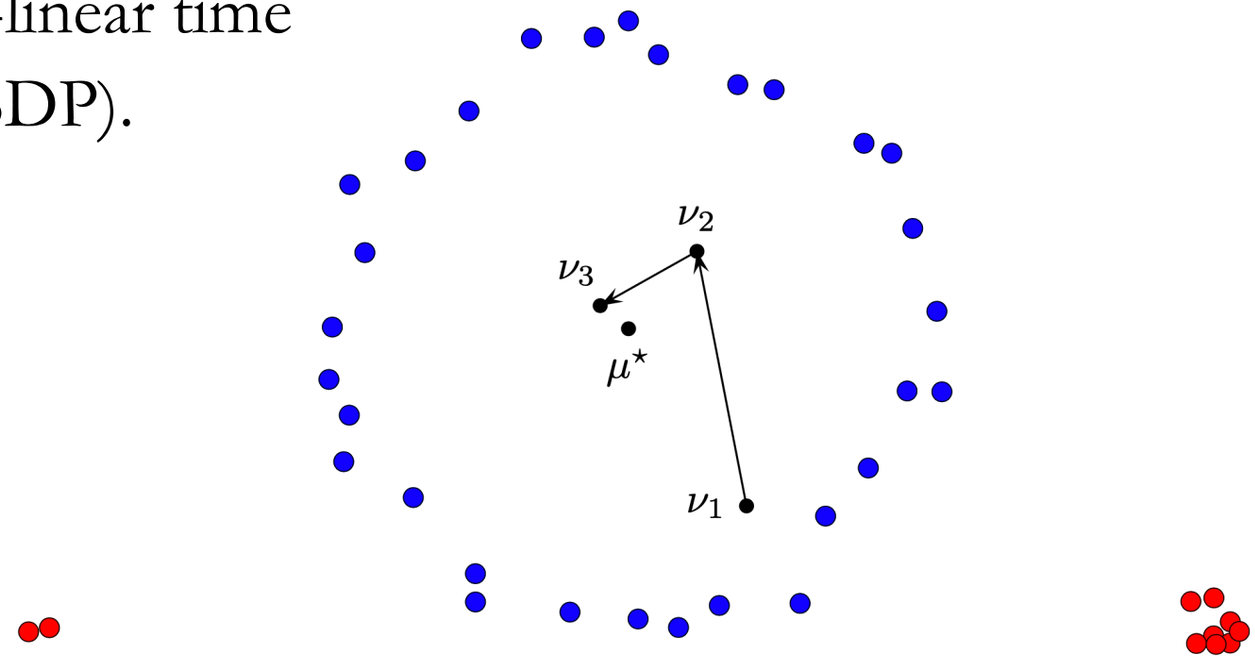


$$\max_{Y \succeq 0, \mathrm{tr}(Y) \le 1} \text{ average of the smallest } (1-\epsilon)\text{−fraction of } \{X_i{}^\top Y\, X_i\}_{i=1}^n$$

optimal solution $\approx yy^\top$, can be written as a packing SDP.

Robust Mean Estimation in Nearly Linear Time
[C Diakonikolas Ge '19]

- Start with coordinate-wise median that is $O(\sqrt{d})$ far from $\mu^\star$.
- The distance to $\mu^\star$ decreases by half in each iteration.
  - Each iteration takes nearly-linear time (mostly solving a packing SDP).

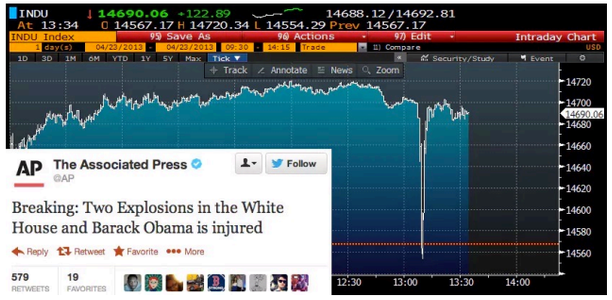


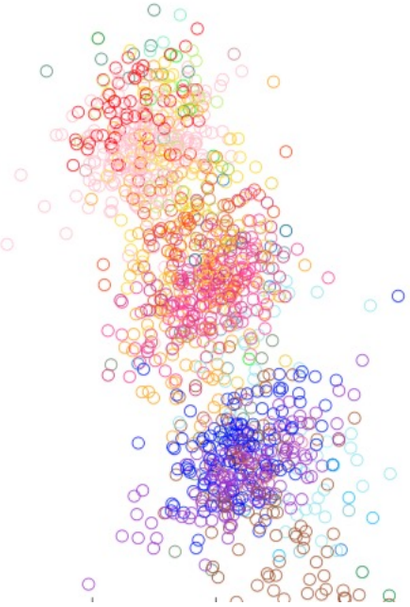
Beyond Mean Estimation

provably robust statistical estimators that are **as efficient as** their non-robust counterparts.

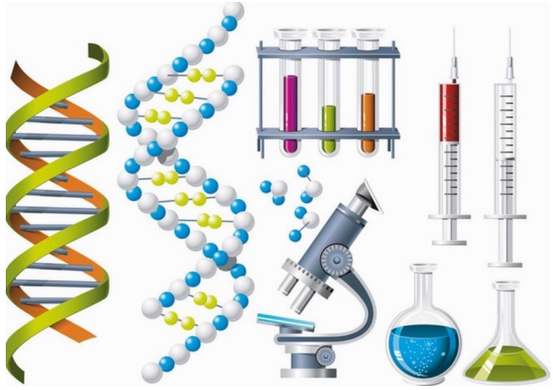
[**C**DG'19]:	mean estimation in time $\tilde{O}(nd)/\mathrm{poly}(\epsilon)$.
[DL'19][DHL'19]:	$\tilde{O}(nd)$ time, heavy-tail mean estimation.
[**C**DGW'19][LY'20]:	covariance estimation.
[C+'20]:	linear regression.
[C+'20][D+'20]:	list-decodable mean estimation.
[**C**L'21]:	learning fixed-structure Bayesian networks.
[DKKLSS'19]:	robust gradient descent.

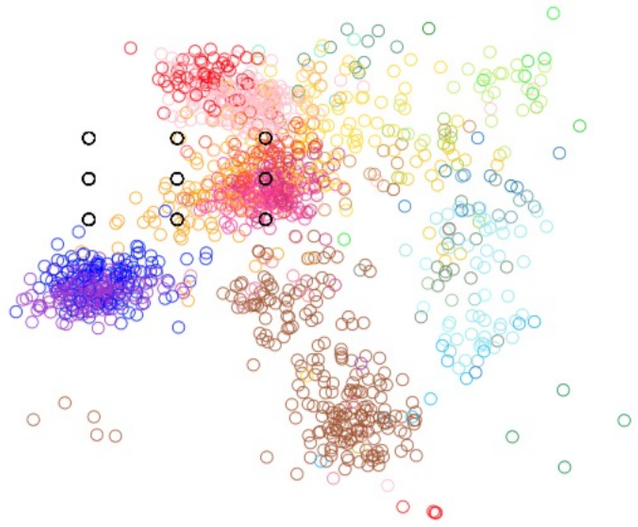
Beyond Mean Estimation



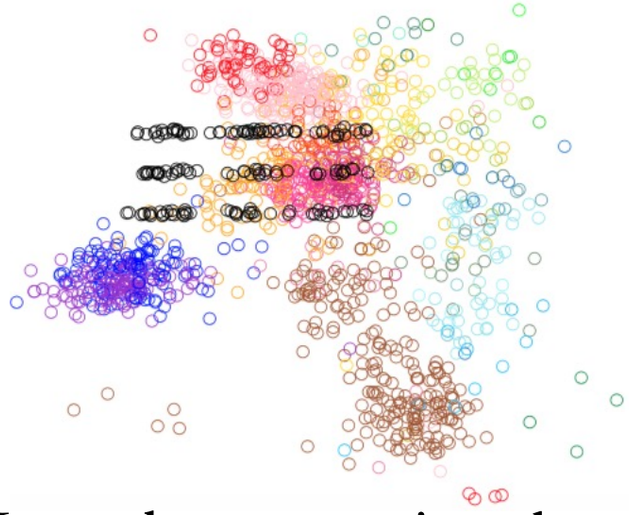


With a few outliers, top singular vectors can be arbitrary.





[Novembre+ ’08]



Use robust top singular vectors.
[DKKLMS ’17]

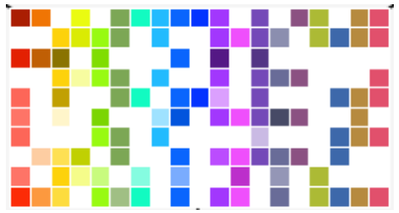
Outline

Part I: Introduction

- Motivation/Challenges

Part II: Robust Algorithms for ML

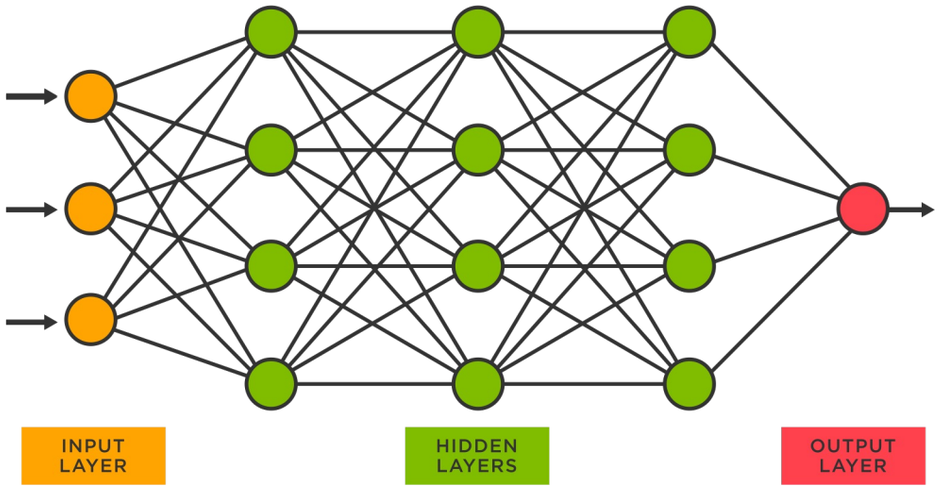
- High-Dimensional Statistics
- Non-Convex Optimization
- Learning with Strategic Agents



Part III: Future Directions

Non-Convex Optimization

Extremely successful in practice.



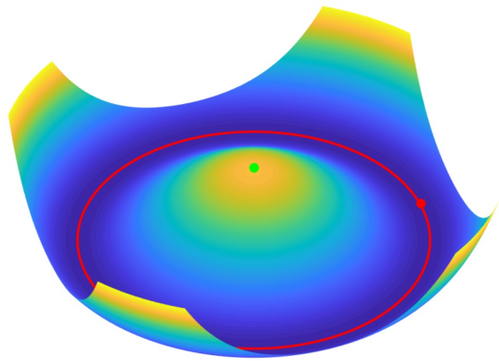
Non-Convex Optimization

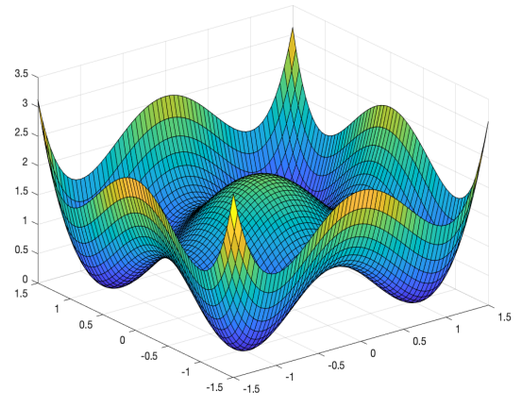
Extremely successful in practice.

- In theory: NP-Hard.
- In practice: can be solved via (stochastic) gradient descent.

Why does non-convex optimization work?

- One possible explanation: **all local optima are globally optimal**.
- **How robust are such landscape results?**





Matrix Completion

An unknown n by n matrix $M^\star$ with rank $r \ll n$.

Input: $M^\star_{ij}$ for a set of observed entries $(i, j) \in \Omega$.

Goal: Recover $M^\star$.

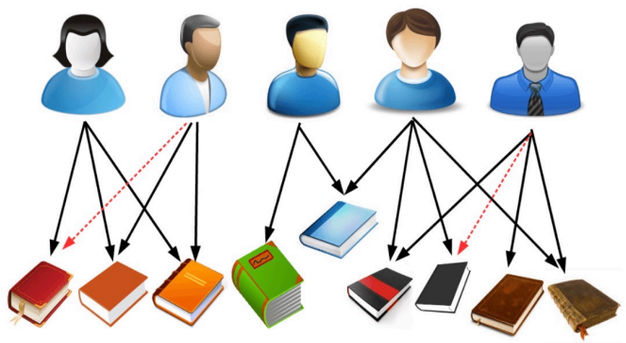
		-1		
			1	
1	1	-1	1	
1				-1
		-1		

1	1	-1	1	-1
1	1	-1	1	-1
1	1	-1	1	-1
1	1	-1	1	-1
1	1	-1	1	-1

Matrix Completion

Application: recommendation systems

(e.g., [Fiat et al. '01, Rennie and Srebro '05, Koren '09]):



Matrix Completion

Application: recommendation systems

(e.g., [Fiat et al. '01, Rennie and Srebro '05, Koren '09]):

n movies

n users

0	-2	-2	-2	2	-2	-2	0
1	1	0	2	-2	1	0	-1
1	0	-1	1	-1	0	-1	-1
2	0	-2	2	-2	0	-2	-2
2	2	0	4	-4	2	0	-2
-1	1	2	0	0	1	2	1
1	1	0	2	-2	1	0	-1
-2	2	4	0	0	2	4	2

$=$

0	-2
1	1
1	0
2	0
2	2
-1	1
1	1
-2	2

1	0	-1	1	-1	0	-1	-1
0	1	1	1	-1	1	1	0

r features

We hope to recover $M^\star$ given $\tilde{O}(nr)$ observations.

Previous Work

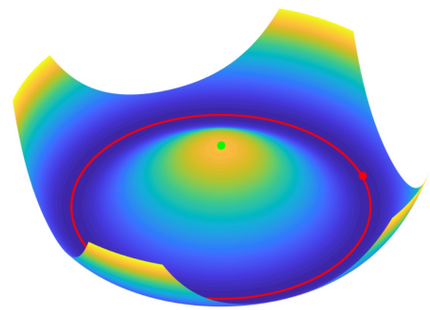
- Convex Relaxation (e.g., [Candès and Tao '10] [Recht '11])
  - min $\|M\|_\star$ s.t. $M_{ij} = M^\star_{ij}$
  - Runtime: $O(n^4)$.

In practice, people use non-convex approaches.

Previous Work

- Non-Convex Approaches $\min \|XX^\top - M^\star\|_F^2$
  - $\min f(X) = \sum_{(i,j)\in\Omega} \left((XX^T)_{ij} - M^\star_{ij} \right)^2$ for $X \in \mathbb{R}^{n\times r}$.

If $\widetilde{\Omega}(nr^6)$ entries are observed **uniformly at random**, any local optima X of f has $XX^\top = M^\star$ [Ge Lee Ma '16].



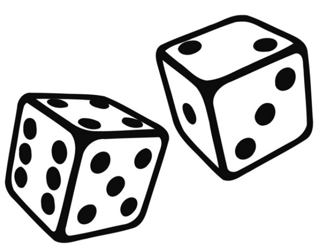
What if the observations are **not** uniformly at random**?**

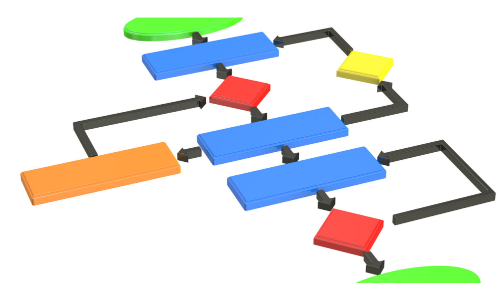
Semi-Random Adversary

0	-2	-2	-2	2	-2	-2	0
1	1	0	2	-2	1	0	-1
1	0	-1	1	-1	0	-1	-1
2	0	-2	2	-2	0	-2	-2
2	2	0	4	-4	2	0	-2
-1	1	2	0	0	1	2	1
1	1	0	2	-2	1	0	-1
-2	2	4	0	0	2	4	2

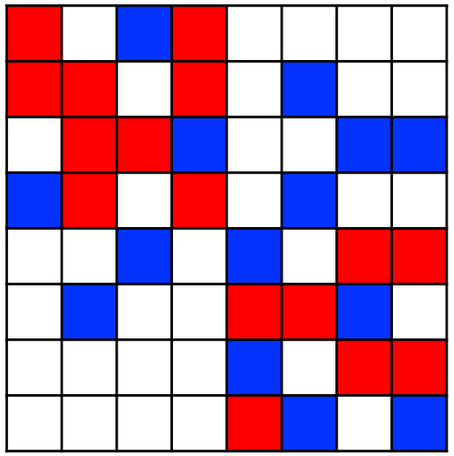
0	-2	-2	-2	2	-2	-2	0
1	1	0	2	-2	1	0	-1
1	0	-1	1	-1	0	-1	-1
2	0	-2	2	-2	0	-2	-2
2	2	0	4	-4	2	0	-2
-1	1	2	0	0	1	2	1
1	1	0	2	-2	1	0	-1
-2	2	4	0	0	2	4	2

0	-2	-2	-2	2	-2	-2	0
1	1	0	2	-2	1	0	-1
1	0	-1	1	-1	0	-1	-1
2	0	-2	2	-2	0	-2	-2
2	2	0	4	-4	2	0	-2
-1	1	2	0	0	1	2	1
1	1	0	2	-2	1	0	-1
-2	2	4	0	0	2	4	2





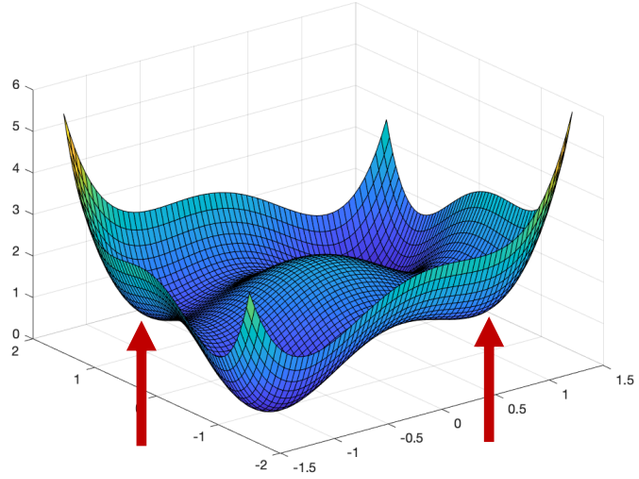
Robust Matrix Completion [C Ge '18]



Does not hurt **convex** approaches:

$$\min \ \|M\|_\star \quad \text{s.t.} \ M_{ij} = M^\star_{ij}$$

However, existing **non-convex** algorithms **are not robust** against such a semi-random adversary.



Counter Examples [C Ge '18]

$f(X) = \sum_{(i,j)\in\Omega} \left(M^{\star}_{ij} - (XX^{\top})_{ij}\right)^2$ has bad local optima.

1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

$X =$

1
1
1
1
-1
-1
-1
-1

1	1	1	1	-1	-1	-1	-1
1	1	1	1	-1	-1	-1	-1
1	1	1	1	-1	-1	-1	-1
1	1	1	1	-1	-1	-1	-1
-1	-1	-1	-1	1	1	1	1
-1	-1	-1	-1	1	1	1	1
-1	-1	-1	-1	1	1	1	1
-1	-1	-1	-1	1	1	1	1

Counter Examples [C Ge '18]

1	1	1	1	-1	-1	-1	-1
1	1	1	1	-1	-1	-1	-1
1	1	1	1	-1	-1	-1	-1
1	1	1	1	-1	-1	-1	-1
-1	-1	-1	-1	1	1	1	1
-1	-1	-1	-1	1	1	1	1
-1	-1	-1	-1	1	1	1	1
-1	-1	-1	-1	1	1	1	1

1
1
1
1
-1
-1
-1
-1

$$\begin{pmatrix}1\\-1\end{pmatrix} \to X = \begin{pmatrix}1\\-(1-\epsilon)\end{pmatrix} \to \cdots \to \begin{pmatrix}1\\1\end{pmatrix}$$

$$XX^\top \approx \begin{pmatrix}1 & -1+\epsilon\\ -1+\epsilon & 1-2\epsilon\end{pmatrix}$$

$$f(X) = \sum_{(i,j)\in\Omega} \left((XX^T)_{ij} - M^\star_{ij}\right)^2 \approx \|XX^\top - M^\star\|_F^2$$

Robust Matrix Completion [C Ge '18]

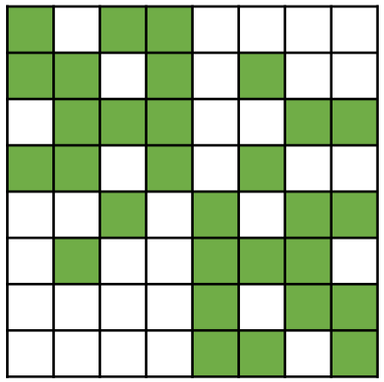
- Against a semi-random adversary:

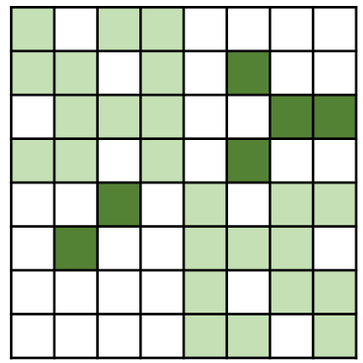
Existing non-convex algorithms are not robust against such a semi-random adversary.

We can fix the non-convex approaches while preserving their efficiency.

Robust Matrix Completion [C Ge '18]

If we know the adversary is changing the input like this …



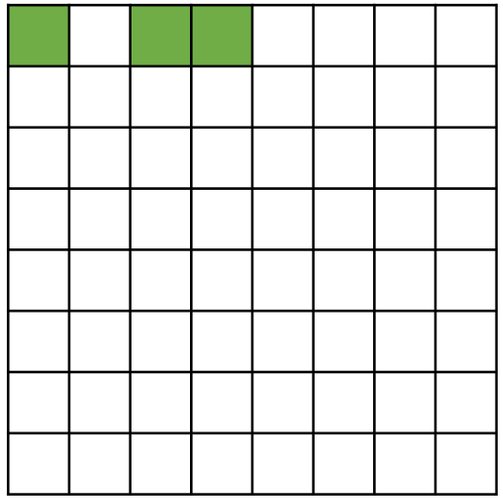


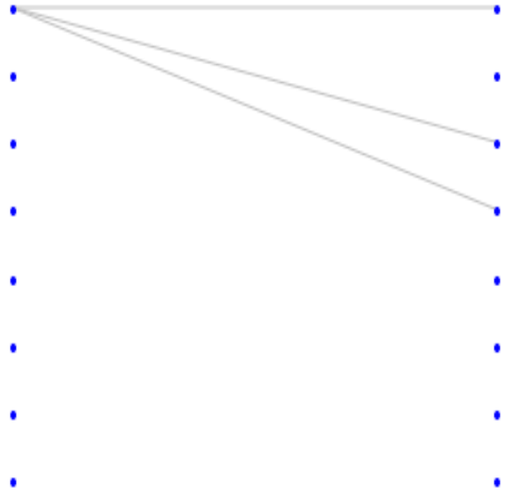
Preprocessing step: reweight the samples so that the input is "similar" to a random input.

$$f_W(X) = \sum_{(i,j)\in\Omega} \boldsymbol{W_{ij}} \left(M^{\star}_{ij} - (XX^{\top})_{ij}\right)^2$$

Our Fix: Preprocessing [C Ge '18]

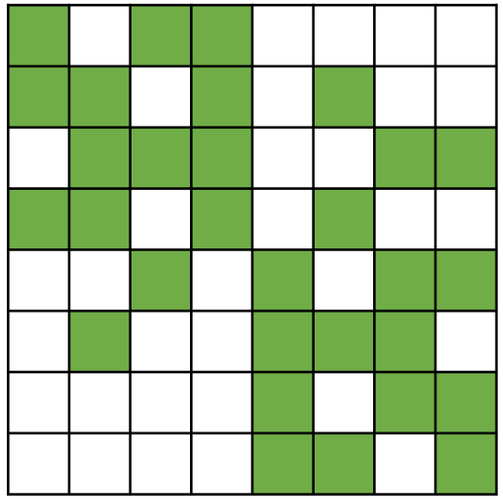
The graph formulation:

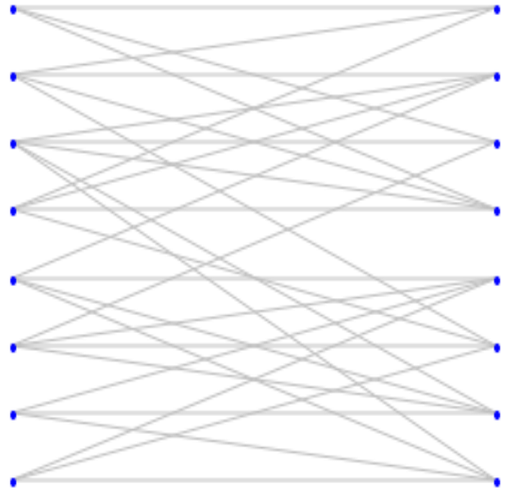




Our Fix: Preprocessing [C Ge '18]

The graph formulation:

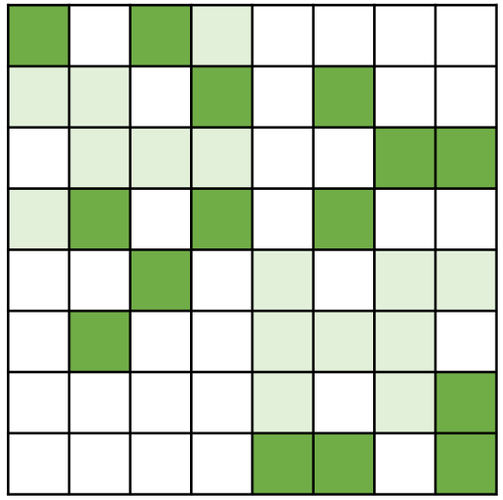


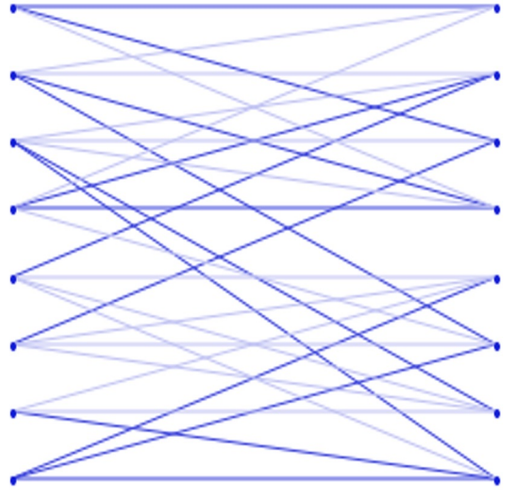


- Input: $H = G(n, n, p)$ + extra edges

Our Fix: Preprocessing [C Ge '18]

The graph formulation:



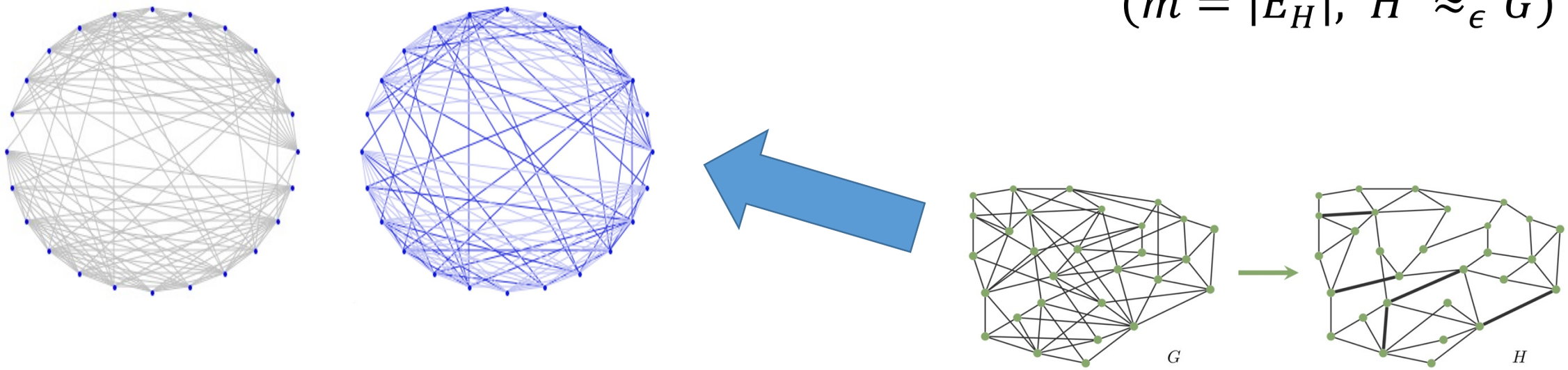


- Input: $H = G(n, n, p)$ + extra edges
- Goal: ~~Recover G.~~ Recover a graph **(spectrally) similar** to G.

Our Fix: Preprocessing [C Ge '18]

We show this can be solved in time: $\tilde{O}(m\ \mathrm{poly}(1/\epsilon))$.

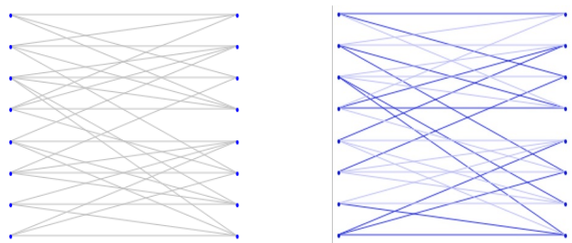
$(m = |E_H|,\ H' \approx_\epsilon G)$

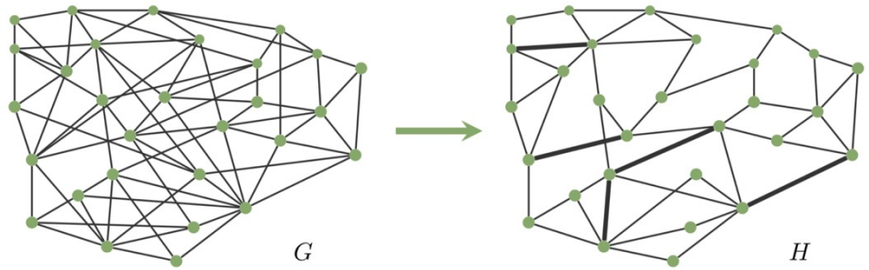


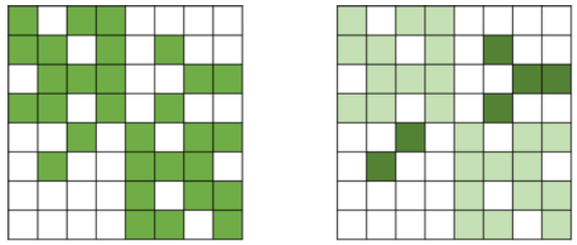
Adapt algorithmic techniques developed for Graph Sparsification

[Batson Spielman Srivastava '12, Lee and Sun '17].

Robust Matrix Completion [C Ge '18]





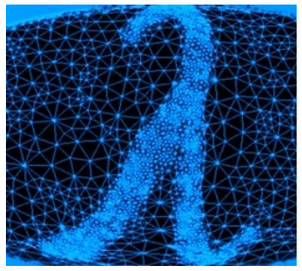


$$f_W(X) = \sum_{(i,j)\in\Omega} \boldsymbol{W_{ij}} \left(M^{\star}_{ij} - (XX^{\top})_{ij}\right)^2$$

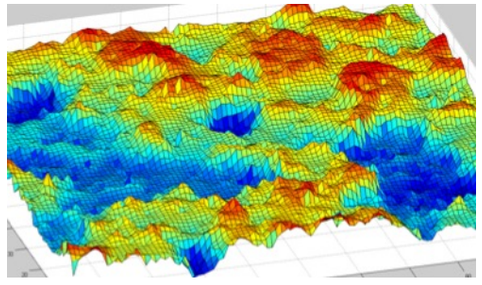
After preprocessing, $f_W(X)$ has no bad local optima.

Robust Non-Convex Optimization

A meta framework for making non-convex approaches robust while preserving their efficiency.



Lightweight convex optimization



Non-convex optimization

Outline

Part I: Introduction

- Motivation/Challenges

Part II: Robust Algorithms for ML

- High-Dimensional Statistics
- Non-Convex Optimization
- Learning with Strategic Agents

Part III: Future Directions

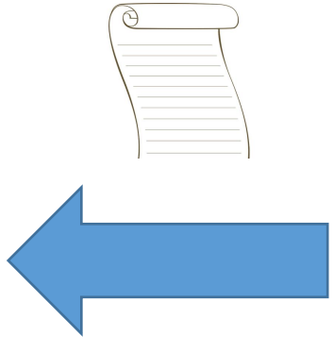
Learning with Strategic Agents

- Data often come from agents, but traditionally we ignore the agents' incentives and consider learning problems in isolation.
- When the algorithms are used to make important decisions about the agents, agents may want to strategically provide data.
- Goal: design optimal algorithms or policies that take the agents' strategic behavior into consideration.

Strategic Reporting [Zhang C Conitzer '19]

Motivating examples:

- Hiring committee



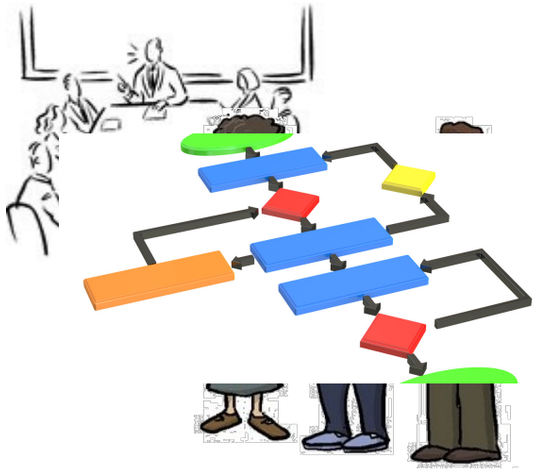
Principal

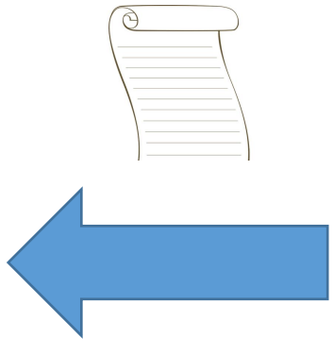
Agent

Strategic Reporting [Zhang C Conitzer '19]

Motivating examples:

- Hiring committee





Principal

Agent

Strategic Reporting [Zhang C Conitzer '19]

Motivating examples:

- Hiring committee
- College scout

The principal does not have direct access to the samples and relies on a strategic agent to provide samples.

What is the optimal policy?

Strategic Reporting [Zhang C Conitzer '19]

Two types of agents: good  or bad  .

- Principal wants to accept the good type and reject the bad type.
- Agent wants to get accepted.

Principal has time to read m paper.

Each agent has n papers.

- Agent generates n samples from a publicly-known distribution based on his/her type, then choose which $m \leq n$ samples to report.

Example 1 [Zhang C Conitzer '19]

$n = 50,\ m = 1.$

A: top conference

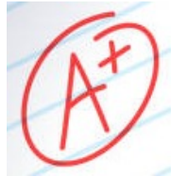
B: average conference

	A	B
	**0.05**	**0.95**
	**0.005**	**0.995**

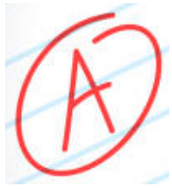
Optimal policy: accept iff agent submits {A}

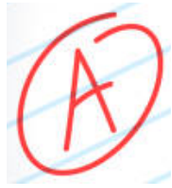
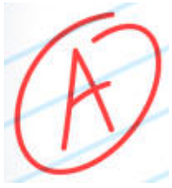
-  is accepted with prob. $1 - 0.95^{50} \approx 0.92.$
-  is accepted with prob. $1 - 0.995^{50} \approx 0.22.$

Example 2 [Zhang **C** Conitzer '19]

			
	**0.8**	**0.2**	**0**
	**0.1**	**0.1**	**0.8**

$m = 2.$

{   }

{   }

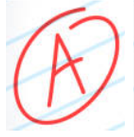
Example 2 [Zhang C Conitzer '19]

$n = 3, m = 2.$

			
	0.8	0.2	0
	0.1	0.1	0.8

Which policy maximizes

Pr[accept | ] − Pr[accept | ]?

Accept {   } and {   }!

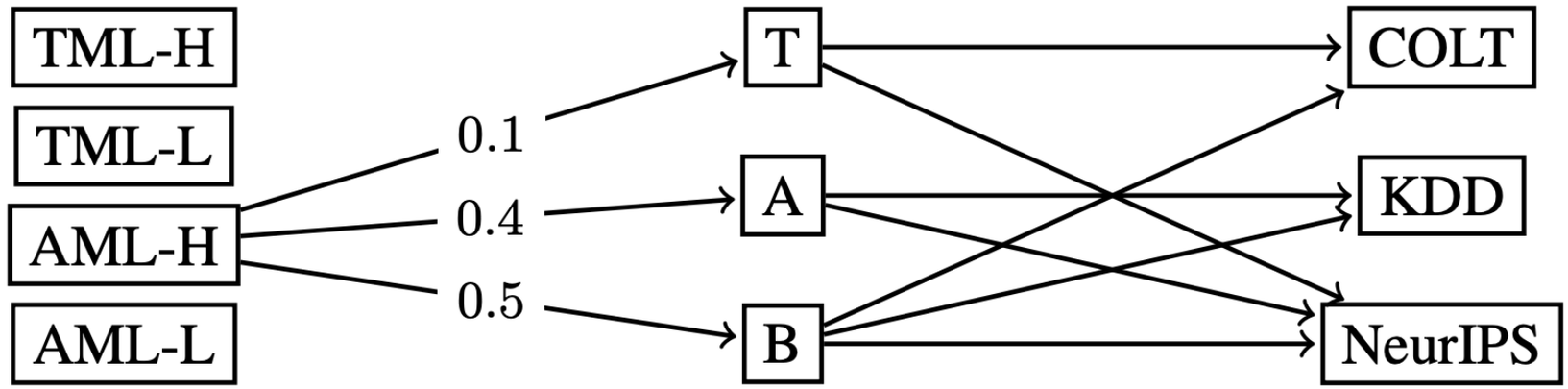
Not monotone in the likelihood ratio!

Automated Mechanism Design

We study the structure and computation of the optimal policy.

- [Zhang **C** Conitzer '19]: Agents can withhold samples.
- [Zhang **C** Conitzer '19]:

  Agents can transform the samples in limited ways.



Automated Mechanism Design

We study the structure and computation of the optimal policy.

- [Zhang **C** Conitzer '19]: Agents can withhold samples.
- [Zhang **C** Conitzer '19]:

  Agents can transform the samples in limited ways.
- [Zhang **C** Conitzer '21]:

  Agents may have different preferences over outcomes.

  Generalization to more outcomes than accept/reject.

Outline

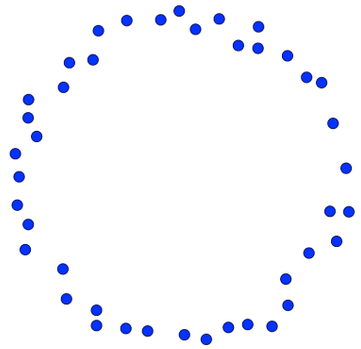
Part I: Introduction

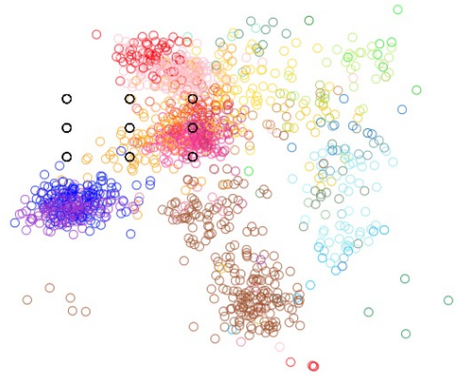
- Motivation/Challenges

Part II: Robust Algorithms for ML

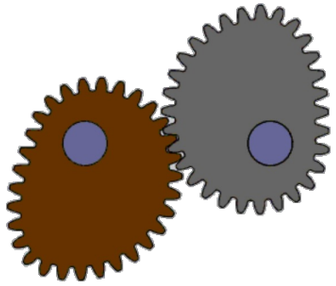
- High-Dimensional Statistics
- Non-Convex Optimization
- Learning with Strategic Agents

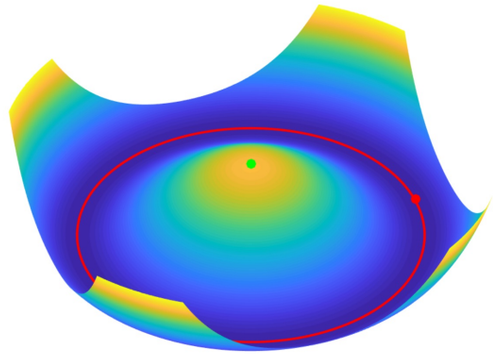
Part III: Future Directions

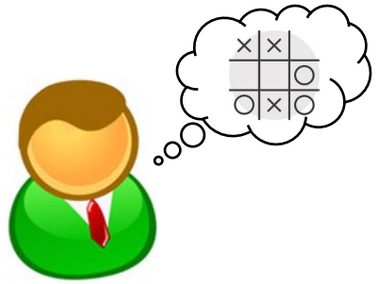


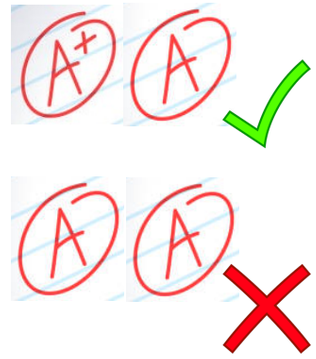


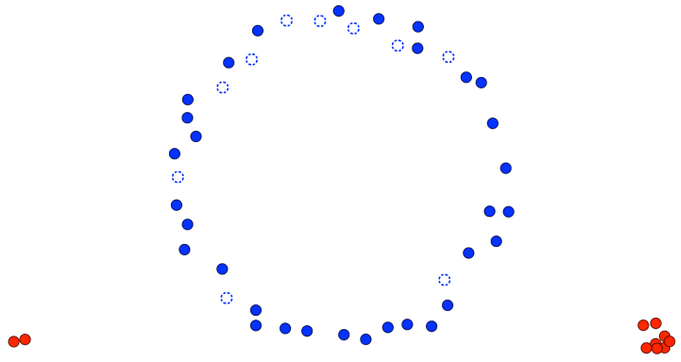
0	-2	-2	-2	2	-2	-2	0
1	1	0	2	-2	1	0	-1
1	0	-1	1	-1	0	-1	-1
2	0	-2	2	-2	0	-2	-2
2	2	0	4	-4	2	0	-2
-1	1	2	0	0	1	2	1
1	1	0	2	-2	1	0	-1
-2	2	4	0	0	2	4	2

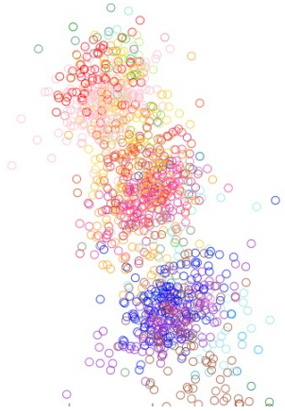


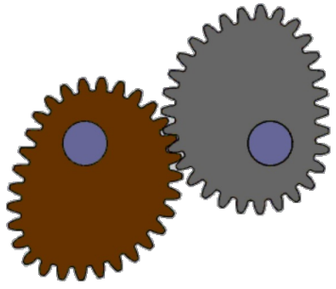




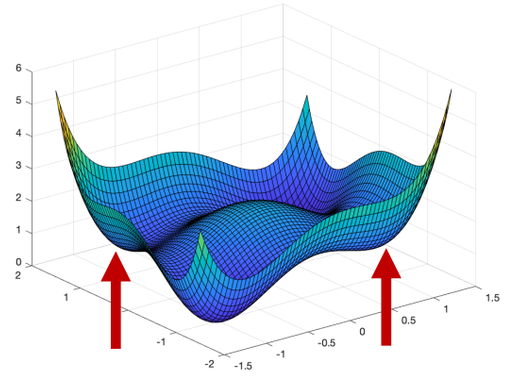


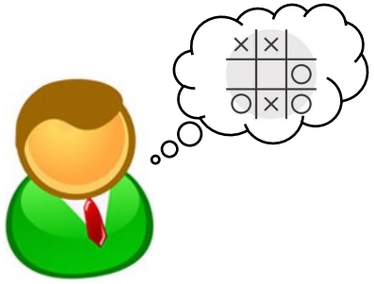


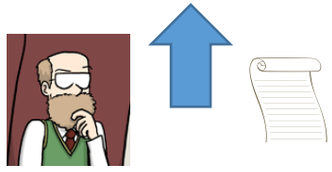


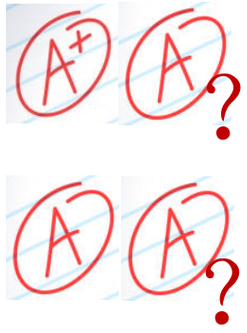


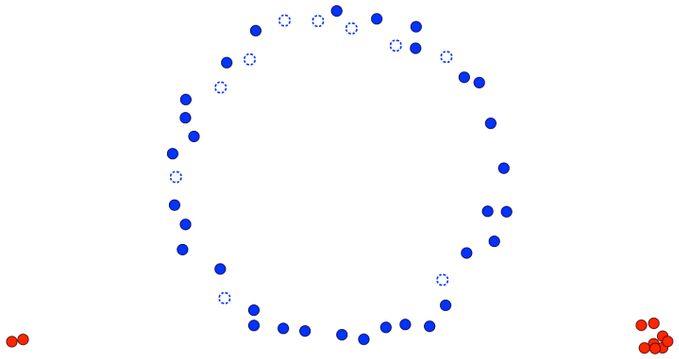
0	-2	-2	-2	2	-2	-2	0
1	1	0	2	-2	1	0	-1
1	0	-1	1	-1	0	-1	-1
2	0	-2	2	-2	0	-2	-2
2	2	0	4	-4	2	0	-2
-1	1	2	0	0	1	2	1
1	1	0	2	-2	1	0	-1
-2	2	4	0	0	2	4	2

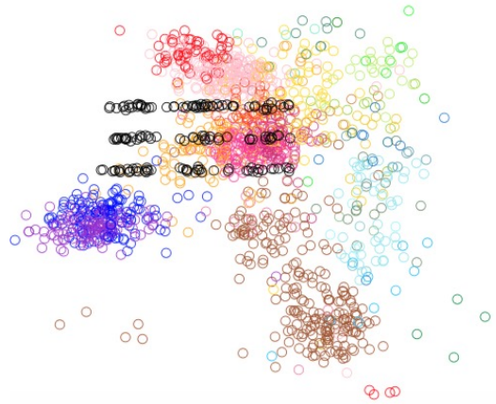


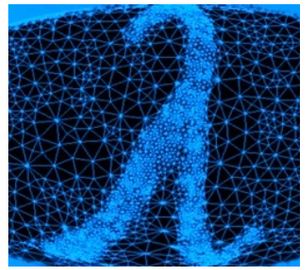


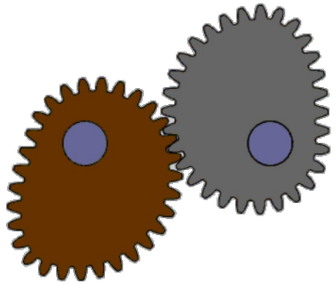




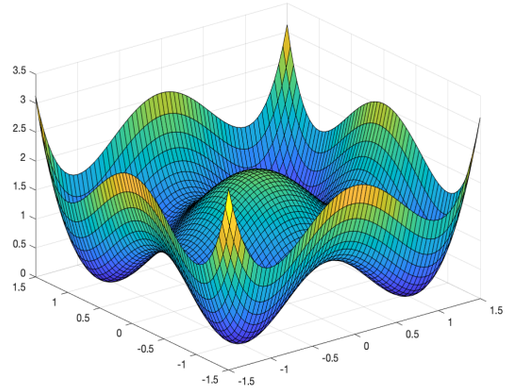


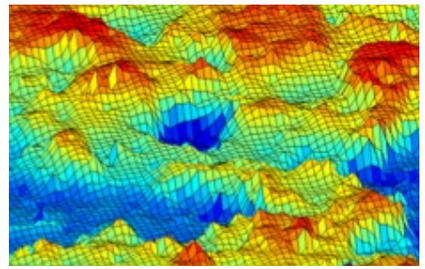


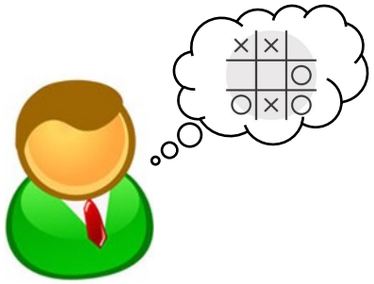


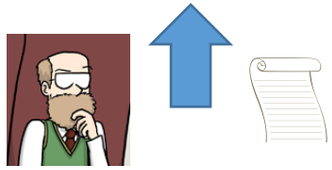


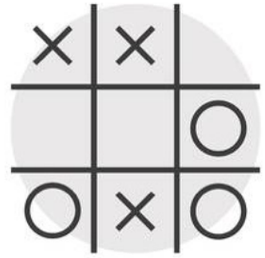
0	-2	-2	-2	2	-2	-2	0
1	1	0	2	-2	1	0	-1
1	0	-1	1	-1	0	-1	-1
2	0	-2	2	-2	0	-2	-2
2	2	0	4	-4	2	0	-2
-1	1	2	0	0	1	2	1
1	1	0	2	-2	1	0	-1
-2	2	4	0	0	2	4	2











Future Direction I: Simple Algorithms

This talk:

- Polynomial time. ✗
- Fastest possible asymptotic runtime. ✓

Future work: speed up the technology transfer into practice.

- Beyond fast asymptotic runtime.
- Leverage modern ML architectures (GPUs, DL packages).

Future Direction I: Simple Algorithms

When a small fraction of the input is corrupted:

[Tukey '60, Huber '64]: provably robust statistical estimators.

[Diakonikolas+ '16, Lai+ '16]: **polynomial-time** robust statistical estimators.

[**C** Diakonikolas Ge '19]:

provably robust estimators that are **as efficient as** their non-robust counterparts.

Fast asymptotic runtime ≠ Practical

Many algorithms are very sophisticated (e.g., ellipsoid method, SoS, JL lemma, matrix multiplication weight update) or require parameters that need careful tuning.

High-Dimensional Robust Statistics

When a small fraction of the input is corrupted:

[Tukey '60, Huber '64]: provably robust statistical estimators.

[Diakonikolas+ '16, Lai+ '16]: **polynomial-time** robust statistical estimators.

[**C** Diakonikolas Ge '19]:

provably robust estimators that are **as efficient as** their non-robust counterparts.

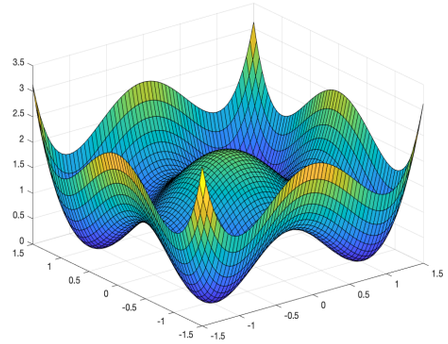
[**C** Diakonikolas Ge Soltanolkotabi '20]:

provably robust estimators that can be computed using **standard first-order methods**.

Robust Mean Estimation via Gradient Descent

[C Diakonikolas Ge Soltanolkotabi '20]

- We consider non-convex formulations of robust mean estimation:
- Despite its non-convexity, we show that any (approximate) stationary point works!



```
for itr = 1:numItr
    Sigma_w_fun = @(v) X' * (w .* (X * v)) - (X' * w)^2 * v;
    [u, lambda] = eigs(Sigma_w_fun, d, 1);
    nabla_f_w = (X * u) .* (X * u) - 2 * (w' * (X * u)) * (X * u);
    w = w - stepSize * nabla_f_w / norm(nabla_f_w);
    w = project_onto_capped_simplex(w, 1 / (N - epsN));
end
```

Robust estimators that are **as efficient as** their non-robust counterparts.

[**C**DG'19]:
mean estimation in time $\tilde{O}(nd)/\mathrm{poly}(\epsilon)$.

[DL'19][DHL'19]:
$\tilde{O}(nd)$ time, heavy-tail mean estimation.

[**C**DGW'19][LY'20]:
covariance estimation.

[C+'20]:
linear regression.

Sparse mean?

[C+'20][D+'20]:
list-decodable mean estimation.

[**C**L'21]:
learning Bayesian networks.

Robust estimators that can be computed using **standard first-order methods**.

[**C**DGS'20]:
mean estimation via gradient descent.

[HLZ'20]:
heavy-tailed mean estimation.
improved # of iterations via mirror descent.

[Zhu+'21]:
covariance estimation, linear regression.

[**C**DKGGS'22, ongoing]:
sparse mean estimation and sparse PCA.

Fast and simple?

Future Direction II: Robust Deep Learning

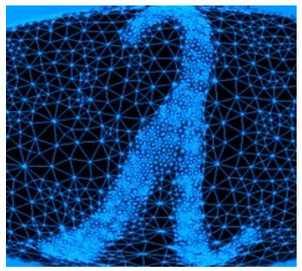
This talk:

- Design and analysis of robust non-convex algorithms for problems that we understand very well w/o corruption.

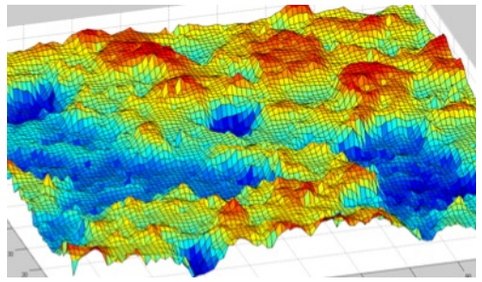
Future work: robustness in deep learning.

- Expand the success of robust non-convex optimization.
- Develop heuristics based on theoretical analysis.

Future Direction II: Robust Deep Learning



Lightweight convex optimization



Non-convex optimization

[**C**GZ, ongoing]: one-bit and noisy matrix completion.

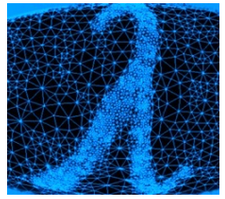
[**C**DG, ongoing]: matrix sensing and phase retrieval.

Future Direction II: Non-Convex Optimization

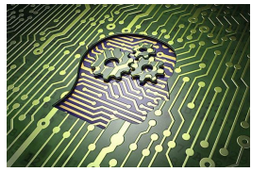
Robustness of neural networks:

- Under what assumptions can neural nets still converge to good parameters if 1% of the data is corrupted?
- If we want to throw away 1% of the training data, which 1% should we throw away?

Spectral Graph Theory



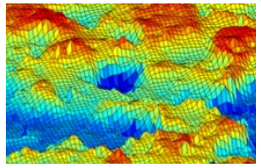
Robustness in Machine Learning



Faster Algorithms for Robust Statistics

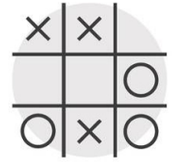
Robust Non-Convex Matrix Completion

Simple Algorithms for Robust Statistics

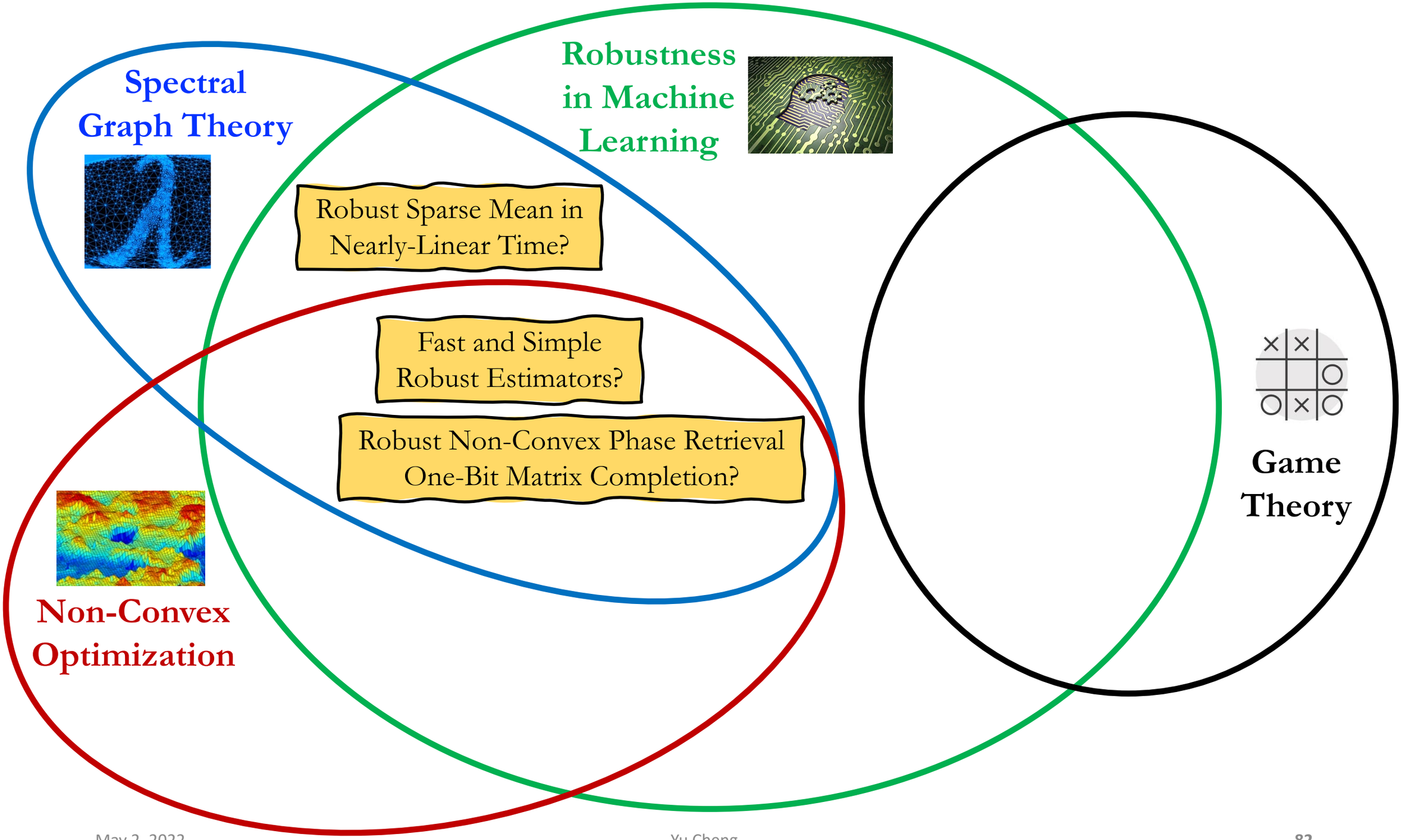


Non-Convex Optimization

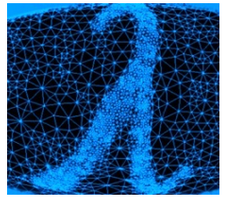
Automated Mechanism Design



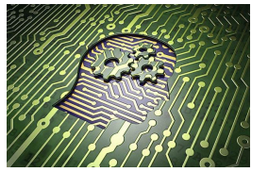
Game Theory


Spectral
Graph Theory
Robustness
in Machine
Learning
Robust Sparse Mean in
Nearly-Linear Time?
Fast and Simple
Robust Estimators?
Robust Non-Convex Phase Retrieval
One-Bit Matrix Completion?
Non-Convex
Optimization
Game
Theory

Spectral Graph Theory



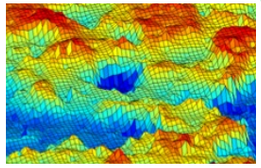
Robustness in Machine Learning



Robust Sparse Mean in Nearly-Linear Time?

Fast and Simple Robust Estimators?

Robust Non-Convex Phase Retrieval
One-Bit Matrix Completion?



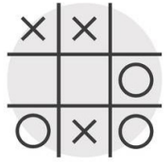
Non-Convex Optimization

Robustly Escaping Saddle Points?

Self-Supervised Learning?

Efficient Algorithms for Planning with Participation?

Private Signaling in Network Routing Games?



Game Theory

Future Directions III: General Theory of Robustness

- Systematic exploration of robustness.
  - Explore different adversarial models and examine whether commonly used algorithms are robust in these models.
  - Develop faster/simpler robust algorithms.
  - Translate the success of robust algorithm design to broader areas.
- Bridge the gap between the growing need for robust algorithms and the lack of general theory of robustness.


UNIVERSITY OF SOUTHERN CALIFORNIA
1880
PALMAM QUI MERUIT FERAT
NSF
SIMONS
FOUNDATION
Duke
UNIVERSITY
INSTITUTE FOR ADVANCED STUDY
TRUTH BEAUTY
FOUNDED 1930
UIC

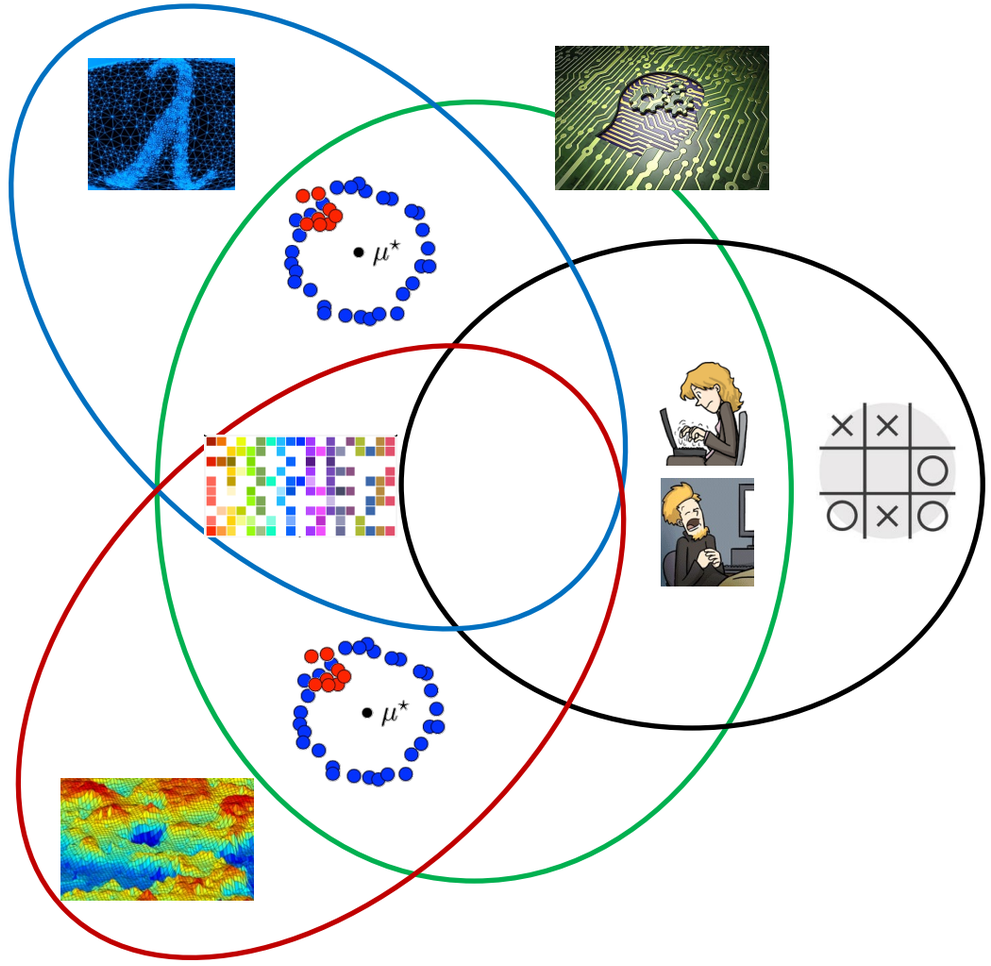
Part I: Introduction

- Motivation/Challenges

Part II: Robust Algorithms for ML

- High-Dimensional Statistics
- Non-Convex Optimization
- Learning with Strategic Agents

Part III: Future Directions



Thanks!
Questions?