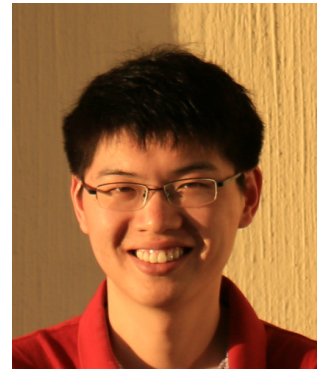
How to Stop Worrying about Ill-Conditioning in Low-Rank Matrix Estimation

Yuejie Chi

Carnegie Mellon University

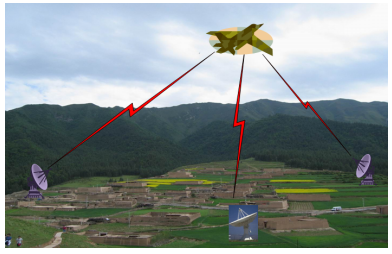
November 2020

Tian Tong
CMU

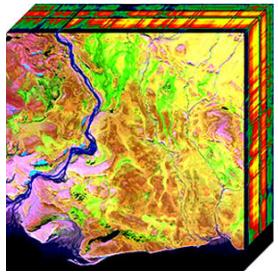


Cong Ma
Berkeley

Low-rank matrices in data science

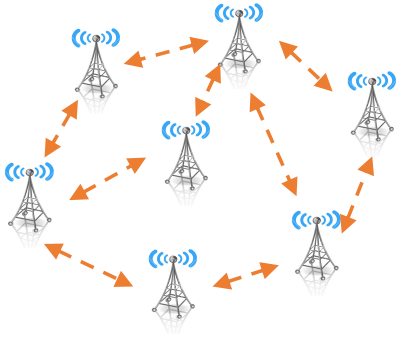


radar imaging

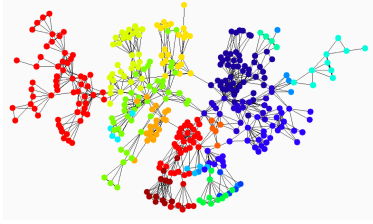


hyperspectral imaging

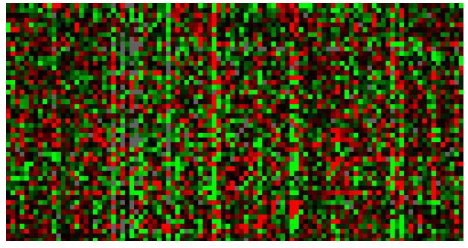
recommendation systems



localization



community detection



bioinformatics

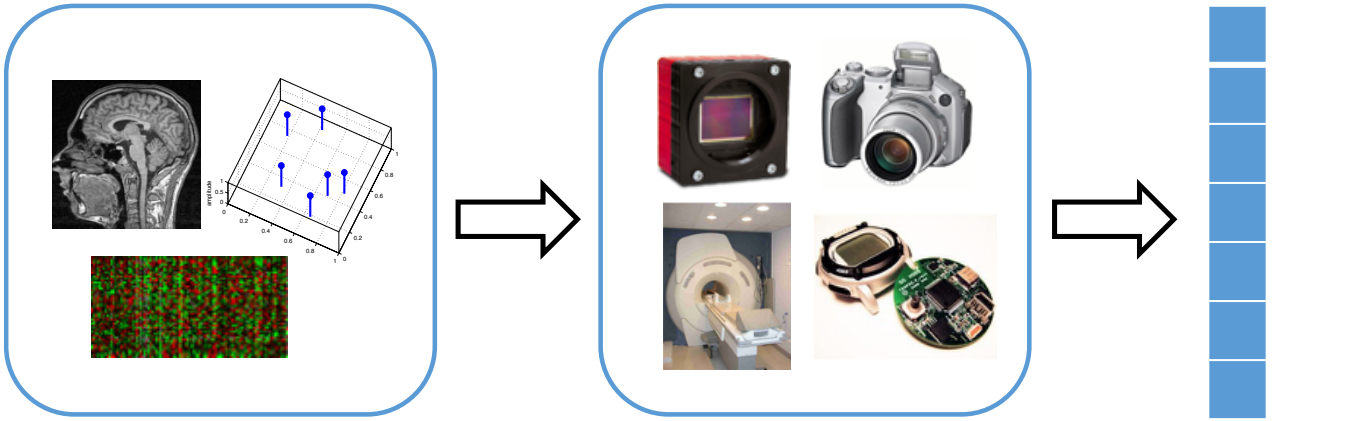
Low-rank matrices are redundant representations of latent information

Low-rank matrix sensing

$\boldsymbol{M} \in \mathbb{R}^{n_1 \times n_2}$
$\operatorname{rank}(\boldsymbol{M}) = r$

$\mathcal{A}(\cdot)$
linear map

$\boldsymbol{y} \in \mathbb{R}^m$



$$\boldsymbol{y} = \mathcal{A}(\boldsymbol{M}) + \text{noise}$$

Recover M in the sample-starved regime:

$$\underbrace{(n_1 + n_2)r}_{\text{degree of freedom}} \lesssim \underbrace{m}_{\text{sensing budget}} \ll \underbrace{n_1 n_2}_{\text{ambient dimension}}$$

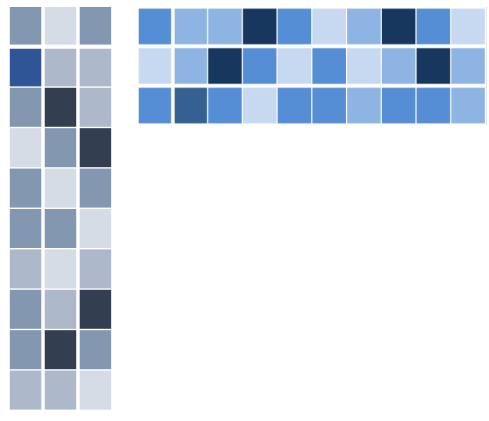
Low-rank matrix factorization

$$\min_{\operatorname{rank}(\boldsymbol{Z})=r} \quad \frac{1}{2}\|\boldsymbol{y}-\mathcal{A}(\boldsymbol{Z})\|_2^2$$

Low-rank matrix factorization

$$\min_{\text{rank}(\boldsymbol{Z})=r} \quad \frac{1}{2}\|\boldsymbol{y}-\mathcal{A}(\boldsymbol{Z})\|_2^2$$

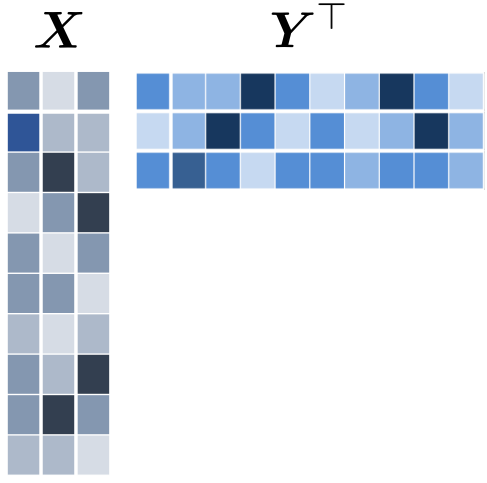
$\Downarrow$

$\boldsymbol{Z} =$ 

Low-rank matrix factorization

$$\min_{\text{rank}(\boldsymbol{Z})=r} \quad \frac{1}{2}\|\boldsymbol{y}-\mathcal{A}(\boldsymbol{Z})\|_2^2$$

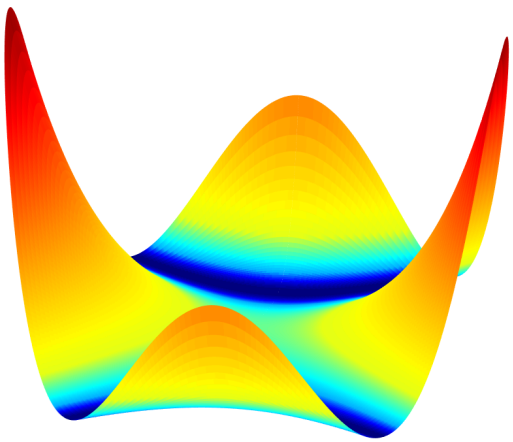
$\Downarrow$

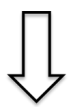
$\boldsymbol{Z} =$ 

$$\min_{\boldsymbol{X},\boldsymbol{Y}} \quad f(\boldsymbol{X},\boldsymbol{Y}) = \frac{1}{2}\left\|\boldsymbol{y}-\mathcal{A}(\boldsymbol{X}\boldsymbol{Y}^\top)\right\|_2^2$$

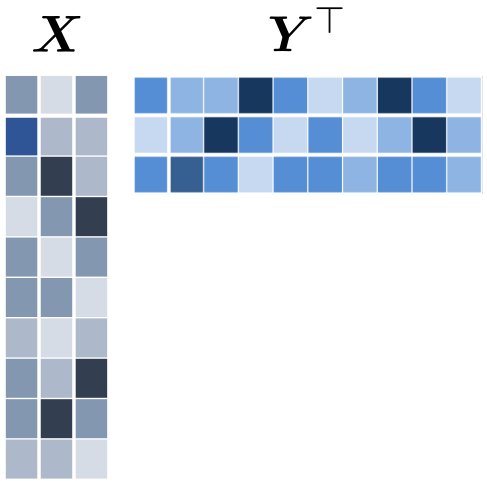
Low-rank matrix factorization

$$\min_{\text{rank}(\boldsymbol{Z})=r} \quad \frac{1}{2}\|\boldsymbol{y}-\mathcal{A}(\boldsymbol{Z})\|_2^2$$





$\boldsymbol{Z} =$

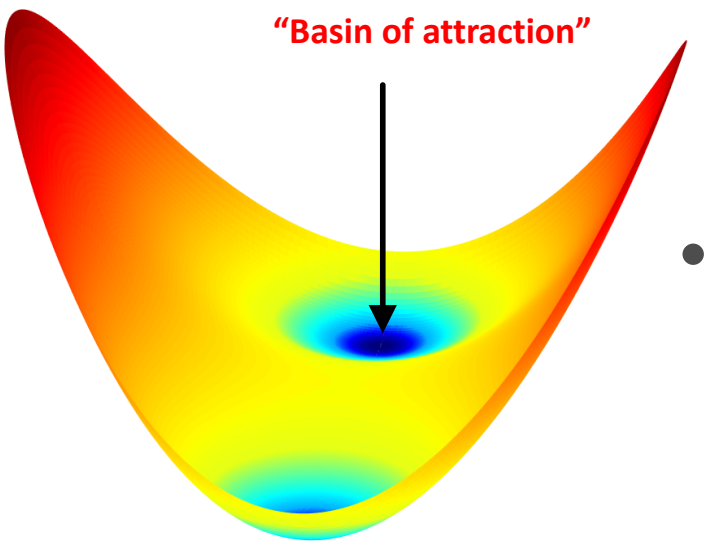


$$\min_{\boldsymbol{X},\boldsymbol{Y}} \quad f(\boldsymbol{X},\boldsymbol{Y}) = \frac{1}{2}\left\|\boldsymbol{y}-\mathcal{A}(\boldsymbol{X}\boldsymbol{Y}^\top)\right\|_2^2$$

Saves memory and computation but introduces nonconvexity!

Prior art: GD with balancing regularization

$$\min_{\boldsymbol{X},\boldsymbol{Y}} \quad f_{\text{reg}}(\boldsymbol{X},\boldsymbol{Y}) = \frac{1}{2}\left\|\boldsymbol{y} - \mathcal{A}(\boldsymbol{X}\boldsymbol{Y}^\top)\right\|_2^2 + \frac{1}{8}\left\|\boldsymbol{X}^\top\boldsymbol{X} - \boldsymbol{Y}^\top\boldsymbol{Y}\right\|_{\text{F}}^2$$



- **Spectral initialization:** find an initial point in the "basin of attraction".

$$(\boldsymbol{X}_0, \boldsymbol{Y}_0) \leftarrow \mathsf{SVD}_r(\mathcal{A}^*(\boldsymbol{y}))$$

- **Gradient iterations:**

$$\boldsymbol{X}_{t+1} = \boldsymbol{X}_t - \eta \nabla_{\boldsymbol{X}} f_{\text{reg}}(\boldsymbol{X}_t, \boldsymbol{Y}_t)$$
$$\boldsymbol{Y}_{t+1} = \boldsymbol{Y}_t - \eta \nabla_{\boldsymbol{Y}} f_{\text{reg}}(\boldsymbol{X}_t, \boldsymbol{Y}_t)$$

for $t = 0, 1, \ldots$

Prior theory for vanilla GD

Theorem (Tu et al., ICML 2016)

Suppose $\boldsymbol{M} = \boldsymbol{X}_\star \boldsymbol{Y}_\star^\top$ *is rank-*r *and has a condition number* $\kappa = \sigma_{\max}(\boldsymbol{M})/\sigma_{\min}(\boldsymbol{M})$. *For low-rank matrix sensing with i.i.d. Gaussian design, vanilla GD (with spectral initialization) achieves*

$$\|\boldsymbol{X}_t \boldsymbol{Y}_t^\top - \boldsymbol{M}\|_{\mathrm{F}} \le \varepsilon \cdot \sigma_{\min}(\boldsymbol{M})$$

- **Computational:** *within* $O\big(\kappa \log \frac{1}{\varepsilon}\big)$ *iterations;*
- **Statistical:** *as long as the sample complexity satisfies*

$$m \gtrsim (n_1 + n_2) r^2 \kappa^2.$$

Prior theory for vanilla GD

Theorem (Tu et al., ICML 2016)

Suppose $\boldsymbol{M} = \boldsymbol{X}_\star \boldsymbol{Y}_\star^\top$ *is rank-*r *and has a condition number* $\kappa = \sigma_{\max}(\boldsymbol{M})/\sigma_{\min}(\boldsymbol{M})$. *For low-rank matrix sensing with i.i.d. Gaussian design, vanilla GD (with spectral initialization) achieves*

$$\|\boldsymbol{X}_t \boldsymbol{Y}_t^\top - \boldsymbol{M}\|_{\mathrm{F}} \le \varepsilon \cdot \sigma_{\min}(\boldsymbol{M})$$

- **Computational:** *within* $O\big(\kappa \log \frac{1}{\varepsilon}\big)$ *iterations;*
- **Statistical:** *as long as the sample complexity satisfies*

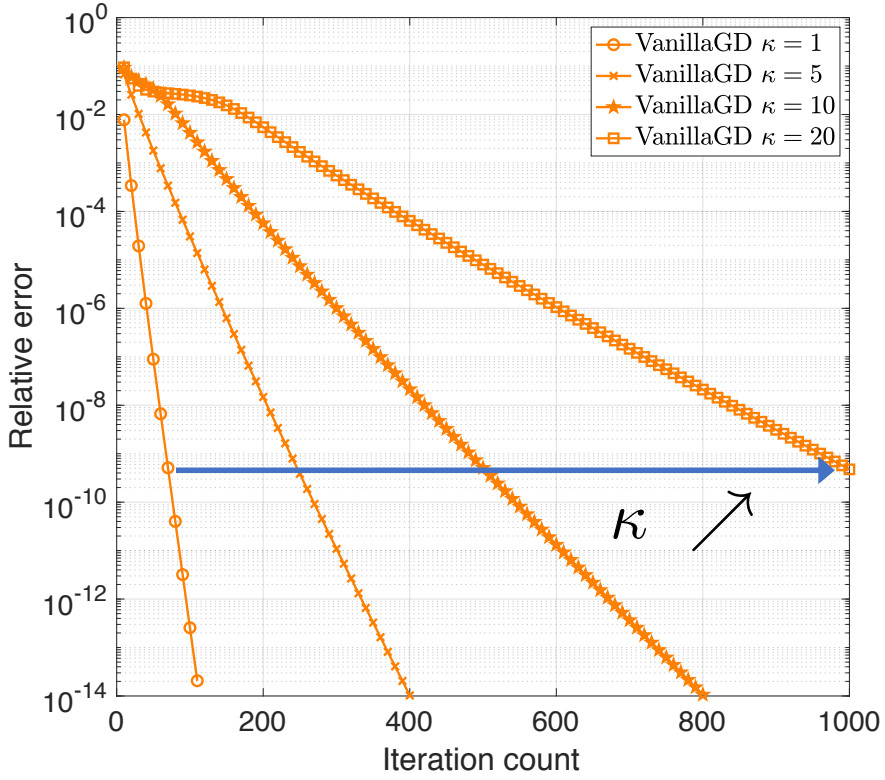
$$m \gtrsim (n_1 + n_2) r^2 \kappa^2.$$

Similar results hold for many low-rank problems.

(Netrapalli et al. '13, Candès, Li, Soltanolkotabi '14, Sun and Luo '15, Chen and Wainwright '15, Zheng and Lafferty '15, Ma et al. '17, ...)

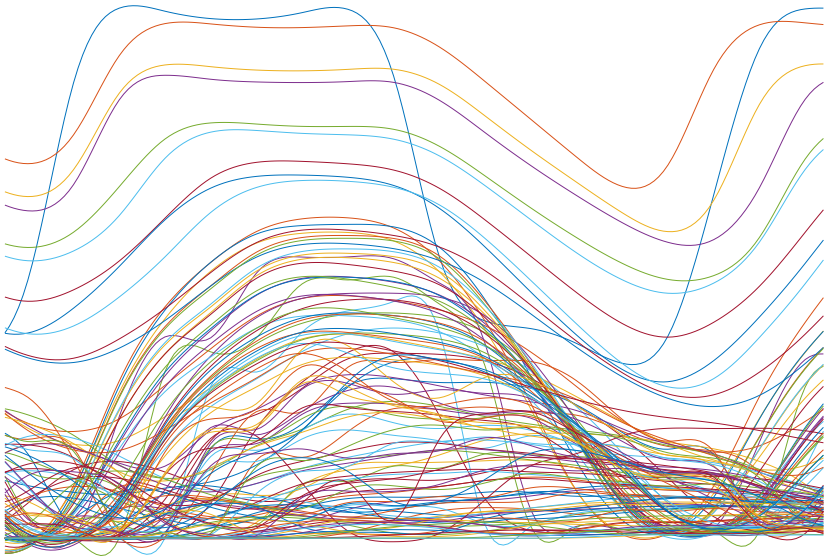
Convergence slows down for ill-conditioned matrices

$$\min_{\boldsymbol{X},\boldsymbol{Y}} \quad f(\boldsymbol{X},\boldsymbol{Y}) = \frac{1}{2}\left\|\mathcal{P}_{\Omega}(\boldsymbol{X}\boldsymbol{Y}^{\top} - \boldsymbol{M})\right\|_{\mathrm{F}}^{2}$$

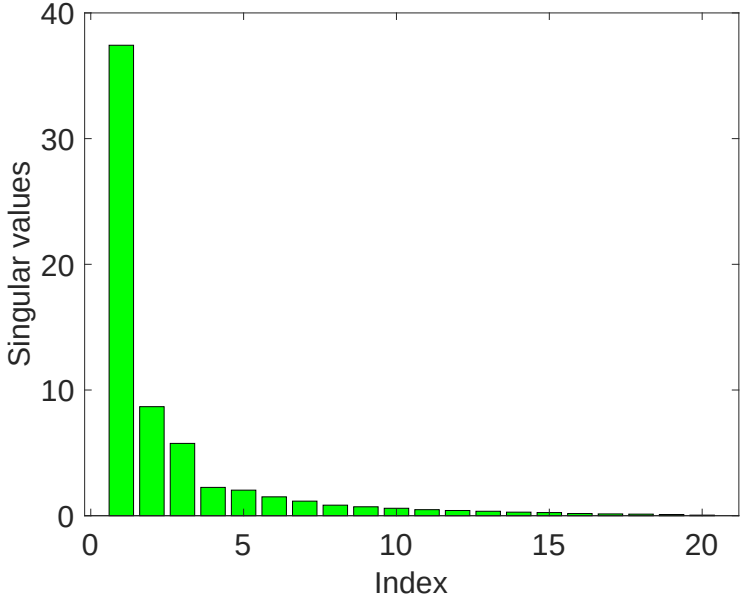


Vanilla GD converges in $O\left(\kappa \log \frac{1}{\varepsilon}\right)$ iterations.

Condition number can be large



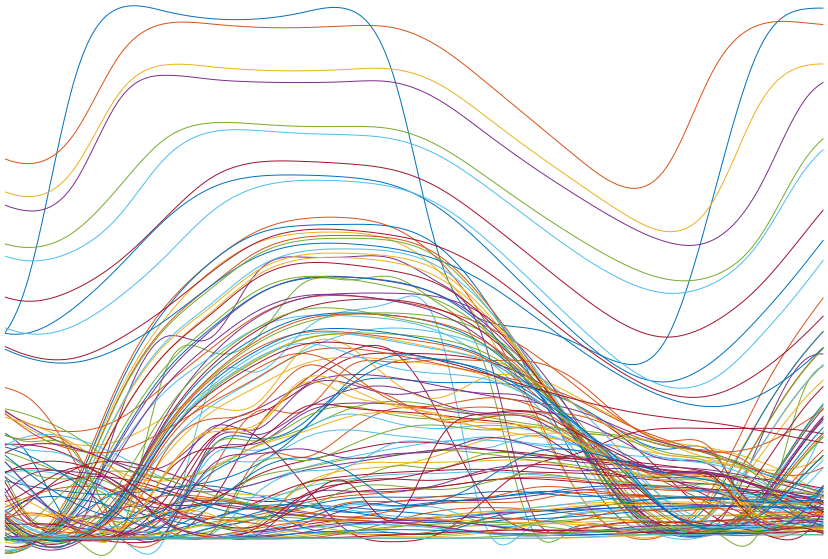
chlorine concentration levels
120 junctions, 180 time slots



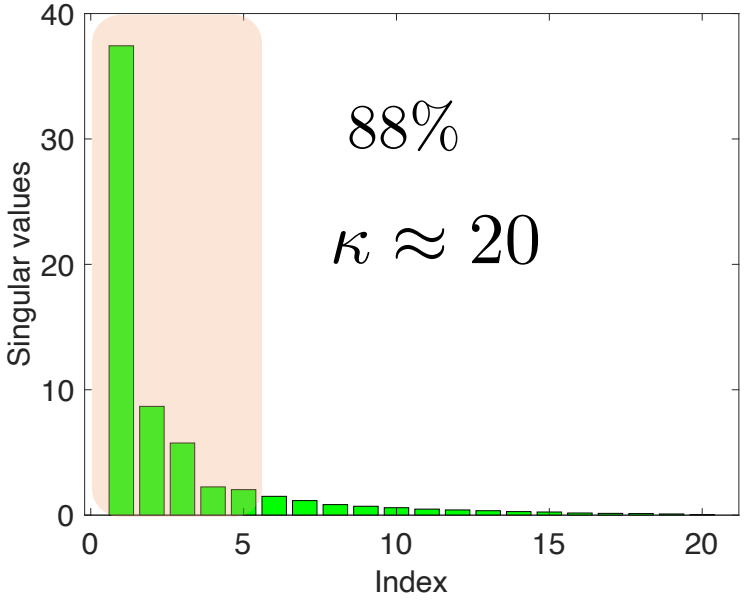
power-law spectrum

Data source: `www.epa.gov/water-research/epanet`

Condition number can be large



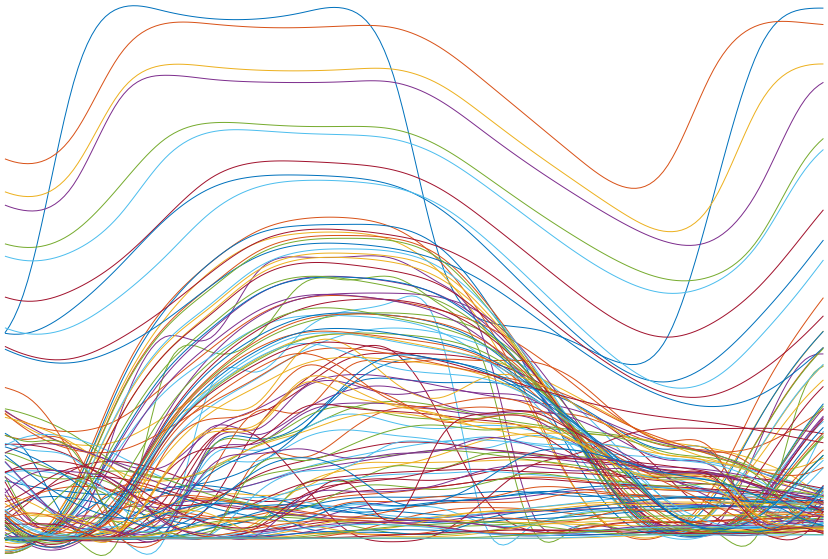
chlorine concentration levels
120 junctions, 180 time slots



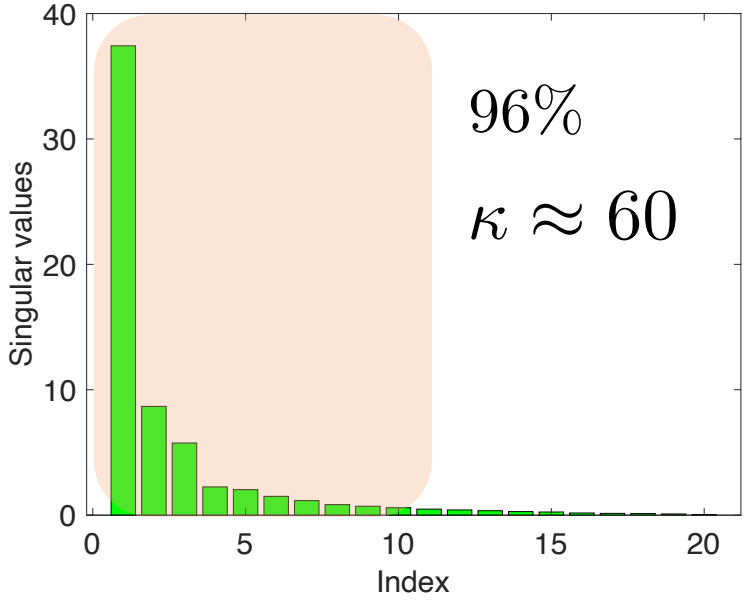
rank-5 approximation

Data source: `www.epa.gov/water-research/epanet`

Condition number can be large



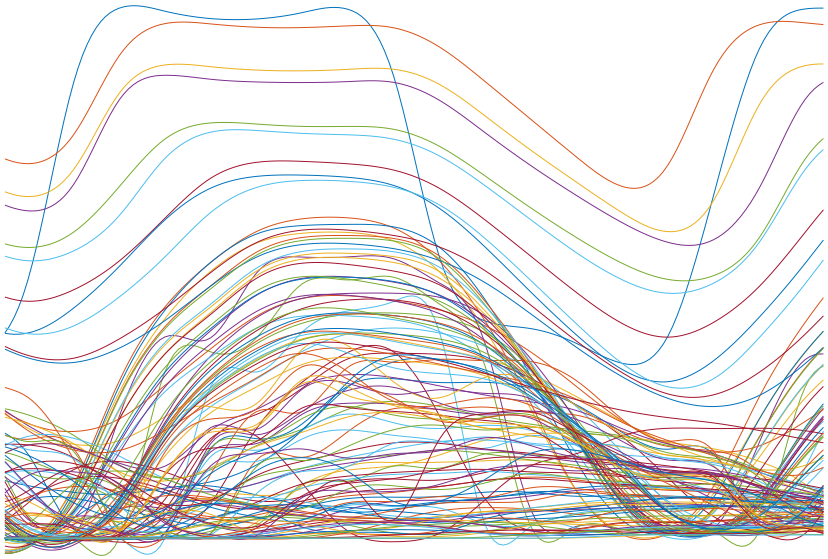
chlorine concentration levels
120 junctions, 180 time slots

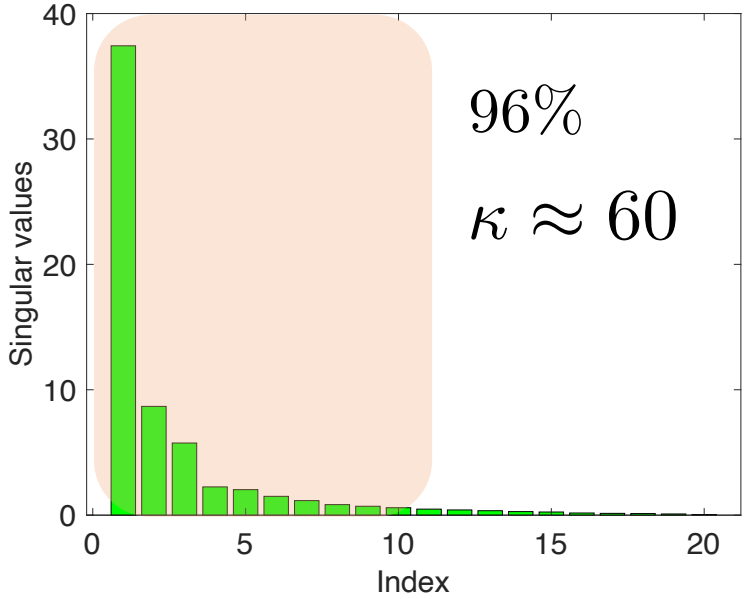


rank-10 approximation

Data source: `www.epa.gov/water-research/epanet`

Condition number can be large





chlorine concentration levels
120 junctions, 180 time slots

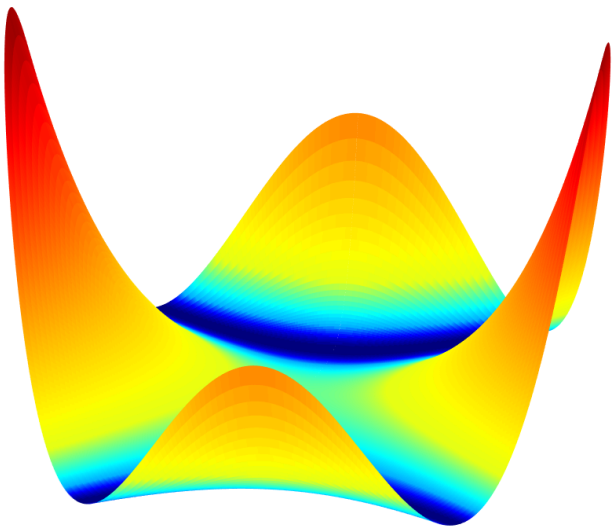
rank-10 approximation

Can we accelerate the convergence rate of GD to $O(\log \frac{1}{\varepsilon})$?

Data source: `www.epa.gov/water-research/epanet`

A new algorithm: scaled gradient descent (ScaledGD)

$$f(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{2} \left\| \boldsymbol{y} - \mathcal{A}(\boldsymbol{X}\boldsymbol{Y}^\top) \right\|_2^2$$



- **Spectral initialization:** find an initial point in the "basin of attraction".
- **Scaled gradient iterations:**

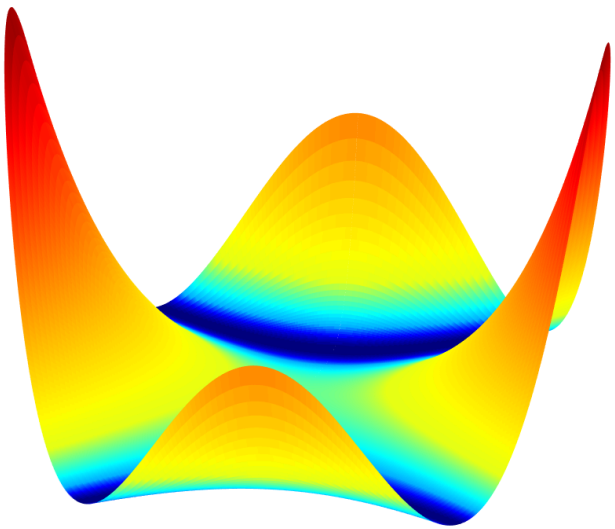
$$\boldsymbol{X}_{t+1} = \boldsymbol{X}_t - \eta \nabla_{\boldsymbol{X}} f(\boldsymbol{X}_t, \boldsymbol{Y}_t) \underbrace{(\boldsymbol{Y}_t^\top \boldsymbol{Y}_t)^{-1}}_{\texttt{preconditioner}}$$

$$\boldsymbol{Y}_{t+1} = \boldsymbol{Y}_t - \eta \nabla_{\boldsymbol{Y}} f(\boldsymbol{X}_t, \boldsymbol{Y}_t) \underbrace{(\boldsymbol{X}_t^\top \boldsymbol{X}_t)^{-1}}_{\texttt{preconditioner}}$$

for $t = 0, 1, \ldots$

A new algorithm: scaled gradient descent (ScaledGD)

$$f(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{2} \left\| \boldsymbol{y} - \mathcal{A}(\boldsymbol{X}\boldsymbol{Y}^\top) \right\|_2^2$$



- **Spectral initialization:** find an initial point in the "basin of attraction".
- **Scaled gradient iterations:**

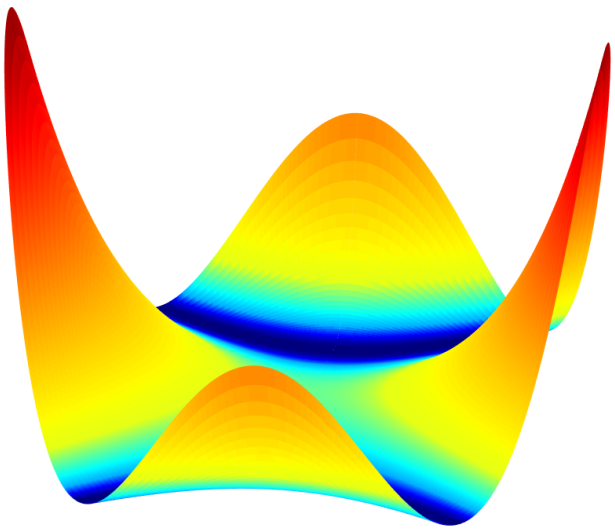
$$\boldsymbol{X}_{t+1} = \boldsymbol{X}_t - \eta \nabla_{\boldsymbol{X}} f(\boldsymbol{X}_t, \boldsymbol{Y}_t) \underbrace{(\boldsymbol{Y}_t^\top \boldsymbol{Y}_t)^{-1}}_{\texttt{preconditioner}}$$

$$\boldsymbol{Y}_{t+1} = \boldsymbol{Y}_t - \eta \nabla_{\boldsymbol{Y}} f(\boldsymbol{X}_t, \boldsymbol{Y}_t) \underbrace{(\boldsymbol{X}_t^\top \boldsymbol{X}_t)^{-1}}_{\texttt{preconditioner}}$$

for $t = 0, 1, \ldots$

A new algorithm: scaled gradient descent (ScaledGD)

$$f(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{2} \left\| \boldsymbol{y} - \mathcal{A}(\boldsymbol{X}\boldsymbol{Y}^\top) \right\|_2^2$$



- **Spectral initialization:** find an initial point in the "basin of attraction".
- **Scaled gradient iterations:**

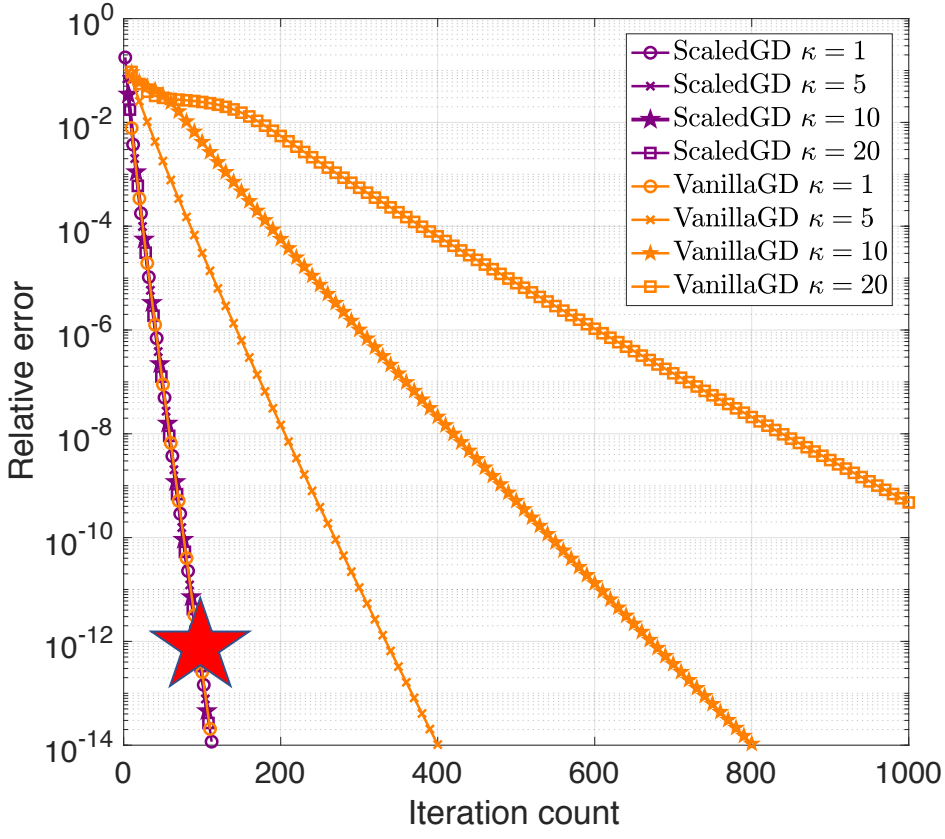
$$\boldsymbol{X}_{t+1} = \boldsymbol{X}_t - \eta \nabla_{\boldsymbol{X}} f(\boldsymbol{X}_t, \boldsymbol{Y}_t) \underbrace{(\boldsymbol{Y}_t^\top \boldsymbol{Y}_t)^{-1}}_{\texttt{preconditioner}}$$

$$\boldsymbol{Y}_{t+1} = \boldsymbol{Y}_t - \eta \nabla_{\boldsymbol{Y}} f(\boldsymbol{X}_t, \boldsymbol{Y}_t) \underbrace{(\boldsymbol{X}_t^\top \boldsymbol{X}_t)^{-1}}_{\texttt{preconditioner}}$$

for $t = 0, 1, \ldots$

ScaledGD is a *preconditioned* gradient method *without* balancing regularization!

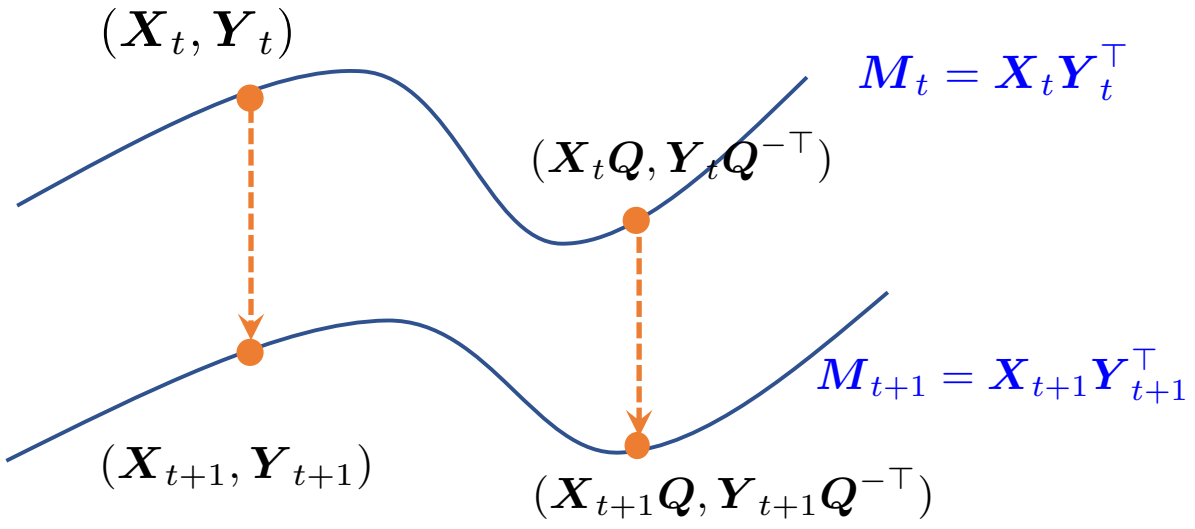
ScaledGD for low-rank matrix completion



Huge computational saving: ScaledGD converges in an κ-independent manner with a minimal overhead!

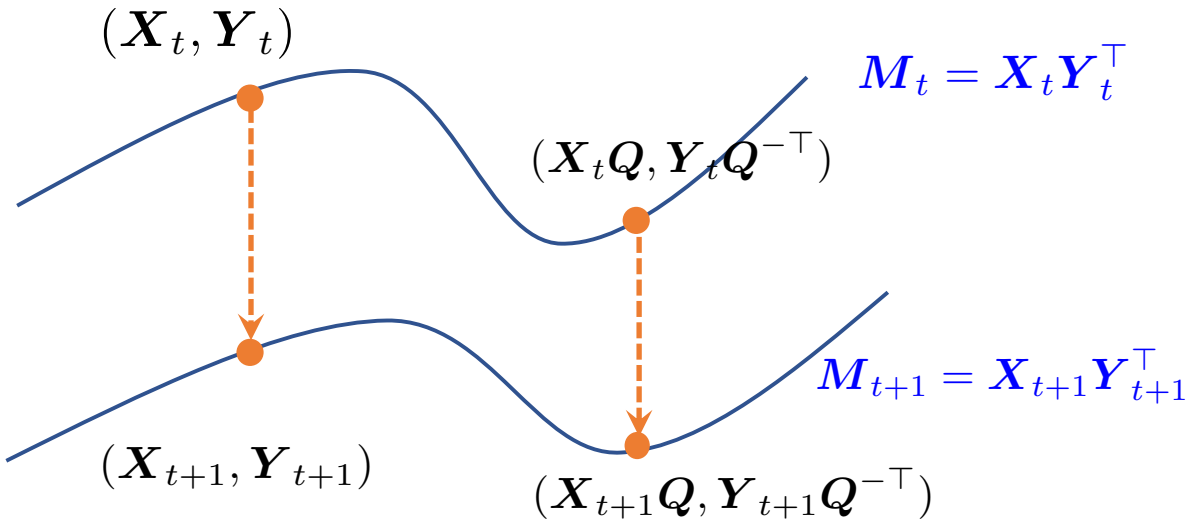
A closer look at ScaledGD

Invariance to invertible transforms: (Tanner and Wei, '16; Mishra '16)



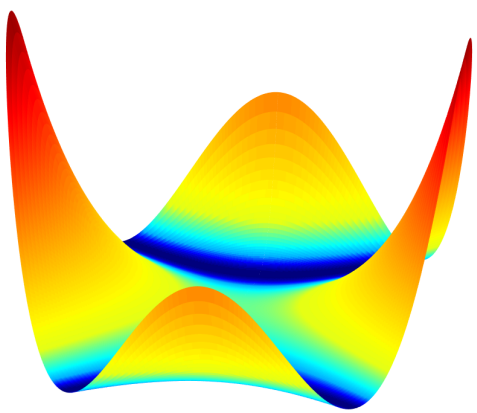
A closer look at ScaledGD

Invariance to invertible transforms: (Tanner and Wei, '16; Mishra '16)



New distance metric as Lyapunov function:

$$\operatorname{dist}^2\left(\begin{bmatrix}\boldsymbol{X}\\ \boldsymbol{Y}\end{bmatrix}, \begin{bmatrix}\boldsymbol{X}_\star\\ \boldsymbol{Y}_\star\end{bmatrix}\right) = \inf_{\boldsymbol{Q}\in \mathrm{GL}(r)} \left\|(\boldsymbol{X}\boldsymbol{Q} - \boldsymbol{X}_\star)\boldsymbol{\Sigma}_\star^{1/2}\right\|_{\mathrm{F}}^2 + \left\|(\boldsymbol{Y}\boldsymbol{Q}^{-\top} - \boldsymbol{Y}_\star)\boldsymbol{\Sigma}_\star^{1/2}\right\|_{\mathrm{F}}^2$$



+ a careful trajectory-based analysis

Theoretical guarantees of ScaledGD

Theorem (Tong, Ma and Chi, 2020)

For low-rank matrix sensing with i.i.d. Gaussian design, ScaledGD with spectral initialization achieves

$$\|\boldsymbol{X}_t\boldsymbol{Y}_t^\top - \boldsymbol{M}\|_{\mathrm{F}} \lesssim \varepsilon \cdot \sigma_{\min}(\boldsymbol{M})$$

- **Computational:** *within* $O\big(\log \frac{1}{\varepsilon}\big)$ *iterations;*
- **Statistical:** *the sample complexity satisfies*

$$m \gtrsim (n_1 + n_2) r^2 \kappa^2.$$

Theoretical guarantees of ScaledGD

Theorem (Tong, Ma and Chi, 2020)

For low-rank matrix sensing with i.i.d. Gaussian design, ScaledGD with spectral initialization achieves

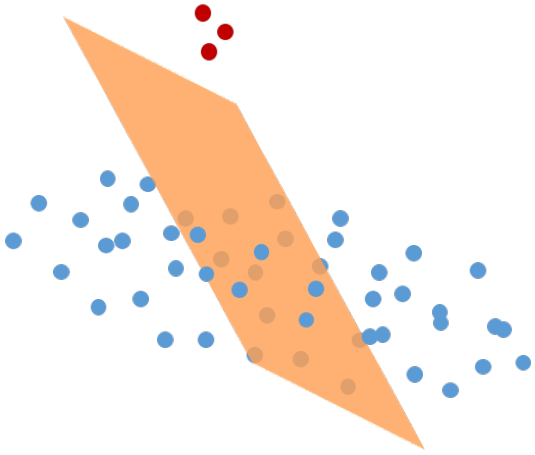
$$\|\boldsymbol{X}_t\boldsymbol{Y}_t^\top - \boldsymbol{M}\|_{\mathrm{F}} \lesssim \varepsilon \cdot \sigma_{\min}(\boldsymbol{M})$$

- **Computational:** *within* $O\big(\log \frac{1}{\varepsilon}\big)$ *iterations;*
- **Statistical:** *the sample complexity satisfies*

$$m \gtrsim (n_1 + n_2) r^2 \kappa^2.$$

Strict improvement over Tu et al.: ScaledGD provably accelerates vanilla GD at the same sample complexity!

ScaledGD works more broadly



$$\begin{bmatrix} \checkmark & ? & ? & ? & \checkmark \\ ? & ? & \checkmark & \checkmark & ? \\ \checkmark & ? & ? & \checkmark & ? \\ ? & ? & \checkmark & ? & ? \\ \checkmark & ? & ? & ? & ? \\ ? & \checkmark & ? & ? & \checkmark \end{bmatrix}$$

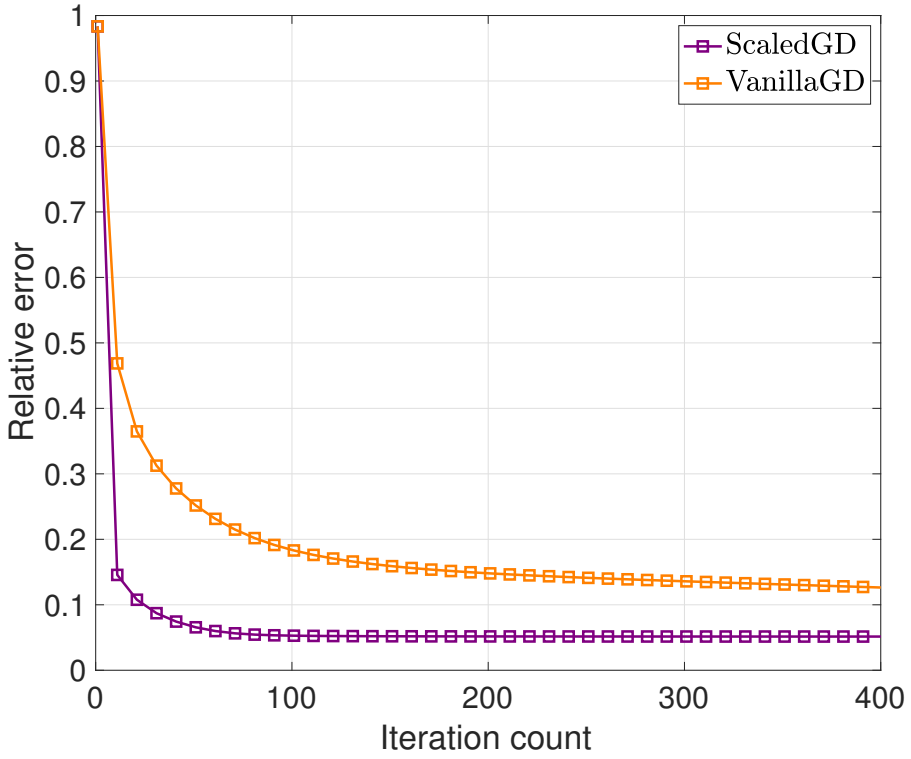
	Robust PCA		**Matrix completion**	
Algorithms	corruption fraction	iteration complexity	sample complexity	iteration complexity
GD	$\frac{1}{\mu r^{3/2}\kappa^{3/2} \vee \mu r \kappa^2}$	$\kappa \log \frac{1}{\varepsilon}$	$(\mu \vee \log n)\mu n r^2 \kappa^2$	$\kappa \log \frac{1}{\varepsilon}$
ScaledGD	$\frac{1}{\mu r^{3/2}\kappa}$	$\log \frac{1}{\varepsilon}$	$(\mu \kappa^2 \vee \log n)\mu n r^2 \kappa^2$	$\log \frac{1}{\varepsilon}$

Huge computation savings at comparable sample complexities!

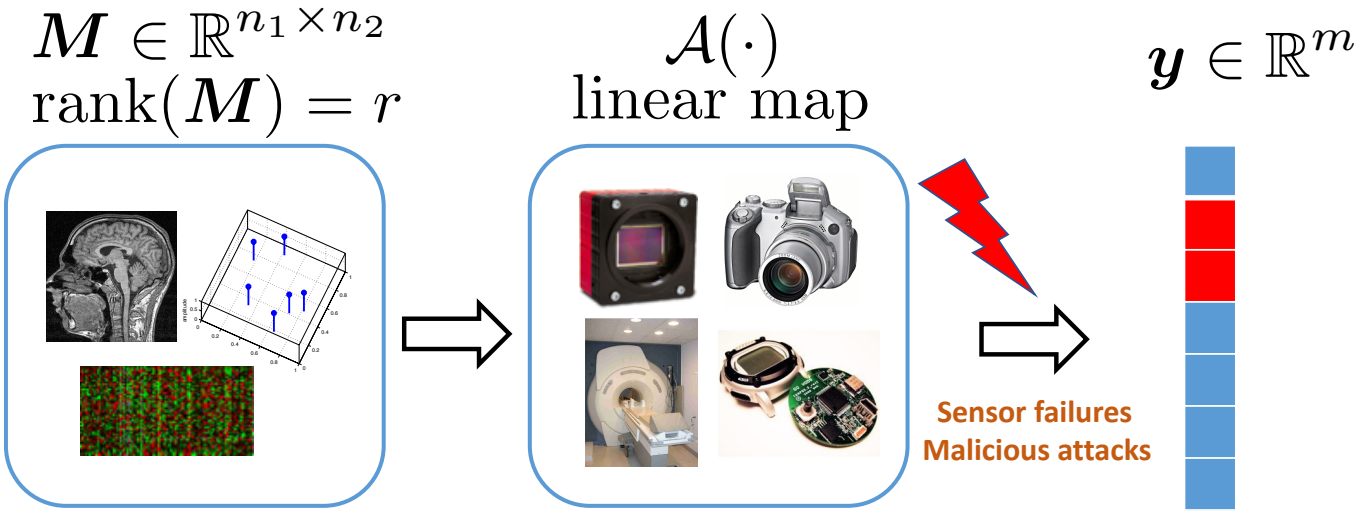
Code available at `https://github.com/Titan-Tong/ScaledGD`

Numerical stability

ScaledGD converges faster than vanilla GD in a small number of iterations (they eventually reach the same accuracy).



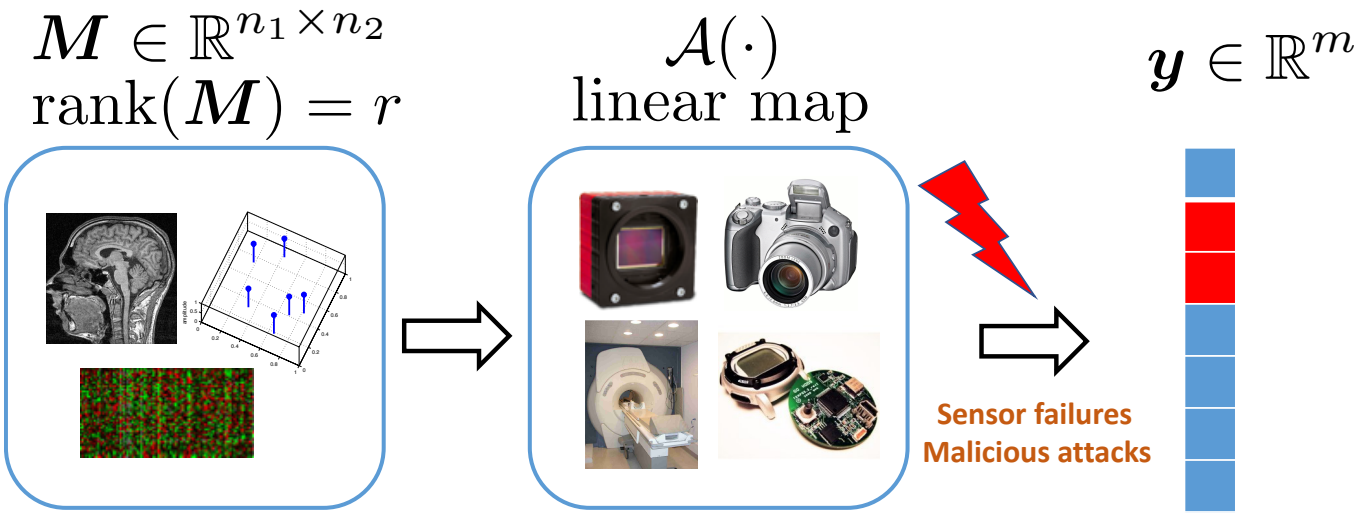
Outlier-corrupted low-rank matrix sensing



$$\boldsymbol{y} \;=\; \mathcal{A}(\boldsymbol{M}) + \underbrace{\text{sparse outliers}}$$

a small fraction (e.g. $p_s \approx 5\%$)

Outlier-corrupted low-rank matrix sensing

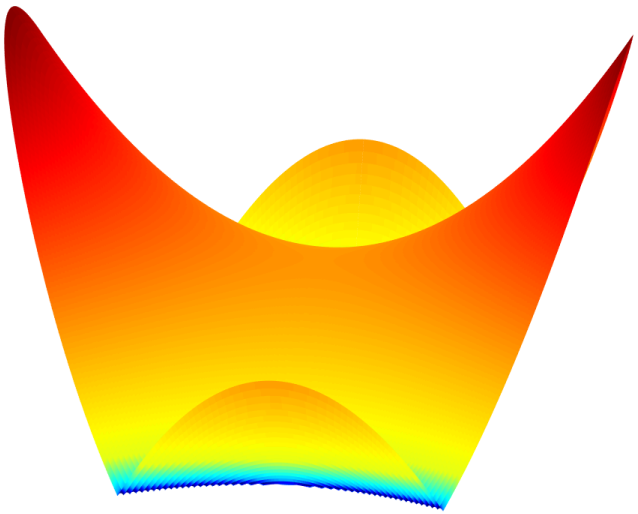


$$\boldsymbol{y} \;=\; \mathcal{A}(\boldsymbol{M}) + \underbrace{\text{sparse outliers}}_{\text{a small fraction (e.g. } p_s \approx 5\%)}$$

Least absolute deviation (LAD)

$$\min_{\boldsymbol{X},\boldsymbol{Y}} \quad f(\boldsymbol{X},\boldsymbol{Y}) = \frac{1}{2}\left\|\boldsymbol{y} - \mathcal{A}(\boldsymbol{X}\boldsymbol{Y}^\top)\right\|_1$$

Scaled subgradient methods



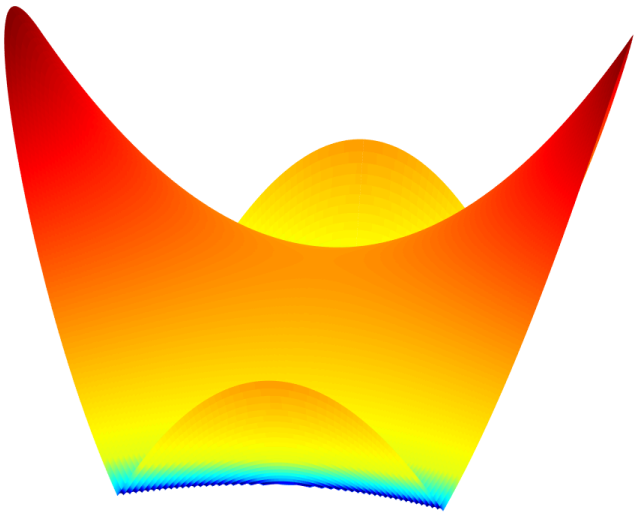
Scaled subgradient iterations:

$$\boldsymbol{X}_{t+1} = \boldsymbol{X}_t - \eta_t \, \partial_{\boldsymbol{X}} f(\boldsymbol{X}_t, \boldsymbol{Y}_t) \underbrace{(\boldsymbol{Y}_t^\top \boldsymbol{Y}_t)^{-1}}_{\texttt{preconditioner}}$$

$$\boldsymbol{Y}_{t+1} = \boldsymbol{Y}_t - \eta_t \, \partial_{\boldsymbol{Y}} f(\boldsymbol{X}_t, \boldsymbol{Y}_t) \underbrace{(\boldsymbol{X}_t^\top \boldsymbol{X}_t)^{-1}}_{\texttt{preconditioner}}$$

where η_t is set as Polyak's or geometric decaying stepsize.

Scaled subgradient methods



Scaled subgradient iterations:

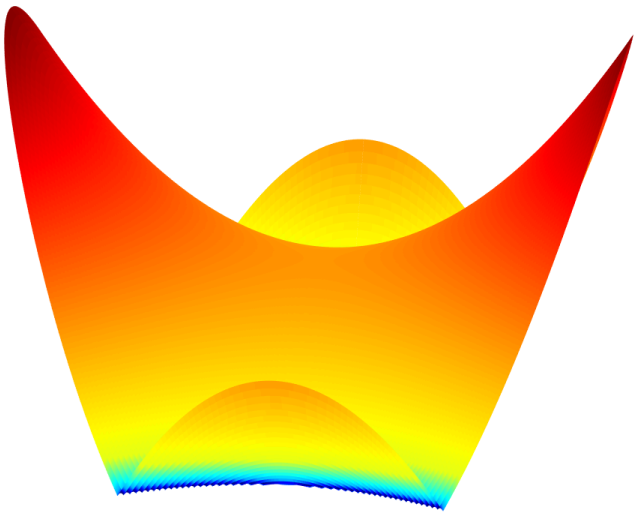
$$\boldsymbol{X}_{t+1} = \boldsymbol{X}_t - \eta_t \, \partial_{\boldsymbol{X}} f(\boldsymbol{X}_t, \boldsymbol{Y}_t) \underbrace{(\boldsymbol{Y}_t^\top \boldsymbol{Y}_t)^{-1}}_{\texttt{preconditioner}}$$

$$\boldsymbol{Y}_{t+1} = \boldsymbol{Y}_t - \eta_t \, \partial_{\boldsymbol{Y}} f(\boldsymbol{X}_t, \boldsymbol{Y}_t) \underbrace{(\boldsymbol{X}_t^\top \boldsymbol{X}_t)^{-1}}_{\texttt{preconditioner}}$$

where η_t is set as Polyak's or geometric decaying stepsize.

	matrix sensing	**quadratic sensing**
Subgradient Method (Charisopoulos et al, '19)	$\frac{\kappa}{(1-2p_s)^2} \log \frac{1}{\varepsilon}$	$\frac{r\kappa}{(1-2p_s)^2} \log \frac{1}{\varepsilon}$
ScaledSM	$\frac{1}{(1-2p_s)^2} \log \frac{1}{\varepsilon}$	$\frac{r}{(1-2p_s)^2} \log \frac{1}{\varepsilon}$

Scaled subgradient methods



Scaled subgradient iterations:

$$\boldsymbol{X}_{t+1} = \boldsymbol{X}_t - \eta_t \, \partial_{\boldsymbol{X}} f(\boldsymbol{X}_t, \boldsymbol{Y}_t) \underbrace{(\boldsymbol{Y}_t^\top \boldsymbol{Y}_t)^{-1}}_{\texttt{preconditioner}}$$

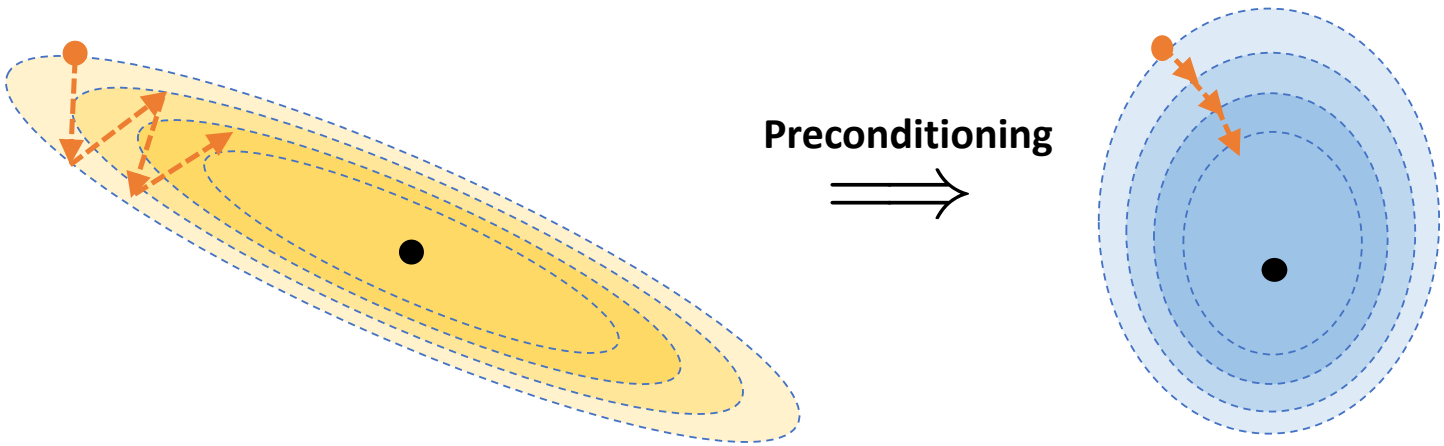
$$\boldsymbol{Y}_{t+1} = \boldsymbol{Y}_t - \eta_t \, \partial_{\boldsymbol{Y}} f(\boldsymbol{X}_t, \boldsymbol{Y}_t) \underbrace{(\boldsymbol{X}_t^\top \boldsymbol{X}_t)^{-1}}_{\texttt{preconditioner}}$$

where η_t is set as Polyak's or geometric decaying stepsize.

	matrix sensing	**quadratic sensing**
Subgradient Method (Charisopoulos et al, '19)	$\frac{\kappa}{(1-2p_s)^2} \log \frac{1}{\varepsilon}$	$\frac{r\kappa}{(1-2p_s)^2} \log \frac{1}{\varepsilon}$
ScaledSM	$\frac{1}{(1-2p_s)^2} \log \frac{1}{\varepsilon}$	$\frac{r}{(1-2p_s)^2} \log \frac{1}{\varepsilon}$

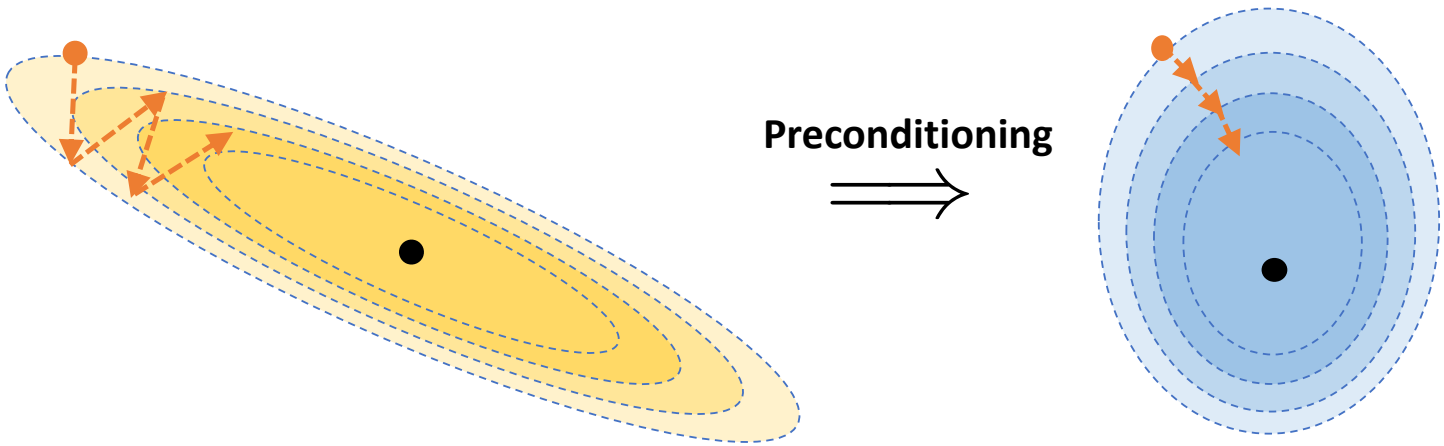
Robustness to both ill-conditioning and adversarial corruptions!

Concluding remarks



Preconditioning dramatically increases the efficiency of vanilla gradient methods even for challenging nonconvex problems!

Concluding remarks



Preconditioning dramatically increases the efficiency of vanilla gradient methods even for challenging nonconvex problems!

Promising directions: unveiling the power of preconditioning in

- Statistical learning
- Reinforcement learning
- Many more ...

Thanks!

- Accelerating Ill-Conditioned Low-Rank Matrix Estimation via **Scaled Gradient Descent**, `arXiv 2005.08898`.
- Low-Rank Matrix Recovery with **Scaled Subgradient Methods**: Fast and Robust Convergence Without the Condition Number, `arXiv 2010.13364`.

https://users.ece.cmu.edu/~yuejiec/