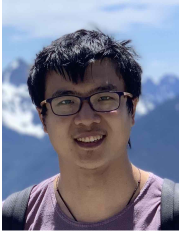
The Power of Preconditioning in Overparameterized Low-Rank Matrix Sensing

Cong Ma

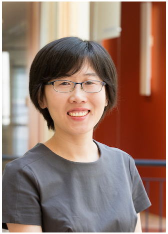
Department of Statistics, UChicago

BDML, Oct. 2023

Xingyu Xu
CMU

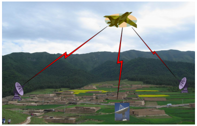


Yandi Shen
UChicago → CMU

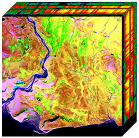


Yuejie Chi
CMU

Low-rank matrices in data science

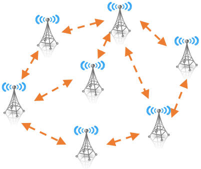


radar imaging

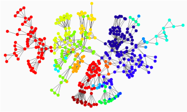


hyperspectral imaging

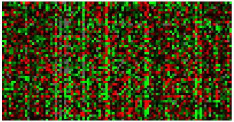
recommendation systems



localization



community detection



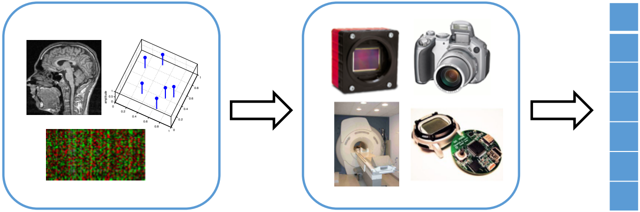
bioinformatics

Low-rank matrix recovery

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

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

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



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

Goal: recover $\boldsymbol{M}$ in the sample-starved regime

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

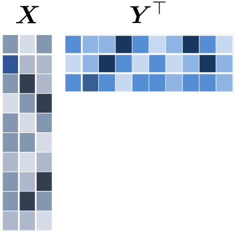
Low-rank matrix factorization

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

Low-rank matrix factorization

$$\min_{\textcolor{blue}{\mathrm{rank}(\boldsymbol{Z})=r}} \quad \frac{1}{2}\left\|\boldsymbol{y}-\mathcal{A}(\boldsymbol{Z})\right\|_2^2$$

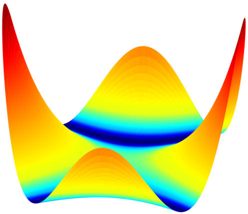
$$\Downarrow \qquad \boldsymbol{Z} =$$



$$\min_{\boldsymbol{X}\in\mathbb{R}^{n_1\times r},\boldsymbol{Y}\in\mathbb{R}^{n_2\times r}} \quad f(\boldsymbol{X},\boldsymbol{Y}) = \frac{1}{2}\left\|\boldsymbol{y}-\mathcal{A}(\boldsymbol{X}\boldsymbol{Y}^\top)\right\|_2^2$$

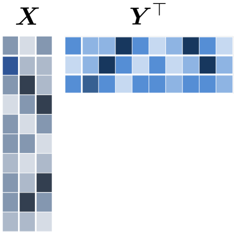
Low-rank matrix factorization

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



$\Downarrow$

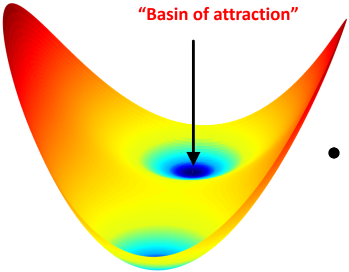
$\boldsymbol{Z} =$



$$\min_{\boldsymbol{X}\in\mathbb{R}^{n_1\times r},\boldsymbol{Y}\in\mathbb{R}^{n_2\times r}} \quad f(\boldsymbol{X},\boldsymbol{Y}) = \frac{1}{2}\left\|\boldsymbol{y}-\mathcal{A}(\boldsymbol{X}\boldsymbol{Y}^\top)\right\|_2^2$$

Prior art: GD with balancing regularization

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



- **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:** for $t = 0, 1, \ldots,$

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

Prior theory for vanilla GD

Condition number $\kappa = \frac{\sigma_{\max}(\boldsymbol{M})}{\sigma_{\min}(\boldsymbol{M})}$

Theorem 1 (Tu et al., ICML 2016)

For low-rank matrix sensing with i.i.d. Gaussian design, vanilla GD (with spectral initialization) achieves

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

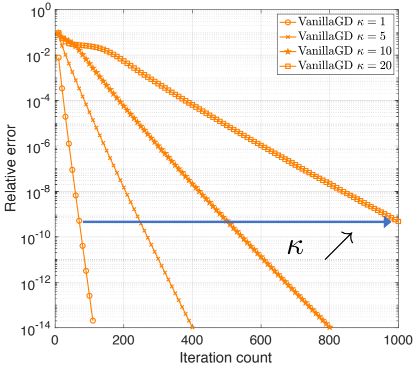
- ***Computational:*** *within* $O(\kappa \log \frac{1}{\varepsilon})$ *iterations;*
- ***Statistical:*** *as long as the sample size satisfies*

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

Similar results hold for many low-rank problems

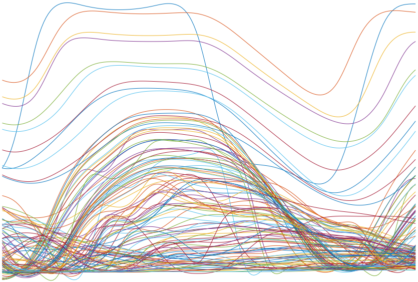
Convergence of vanilla gradient descent

Condition number $\kappa = \frac{\sigma_{\max}(\boldsymbol{M})}{\sigma_{\min}(\boldsymbol{M})}$

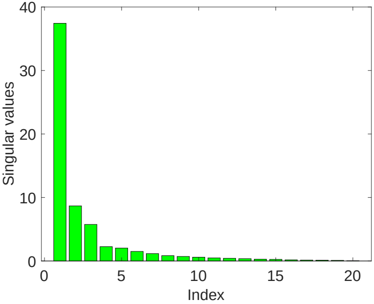


Vanilla GD converges in $O\left(\kappa \log \frac{1}{\varepsilon}\right)$ iterations

Condition number can be large



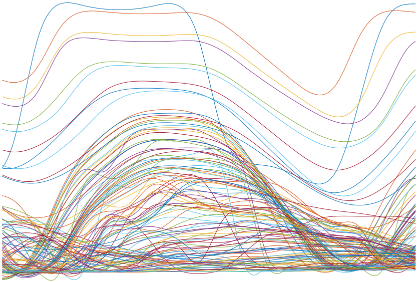
chlorine concentration levels
120 junctions, 180 time slots



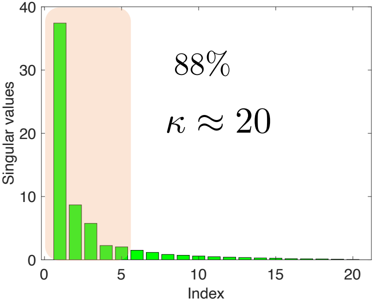
power-law spectrum

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

Condition number can be large



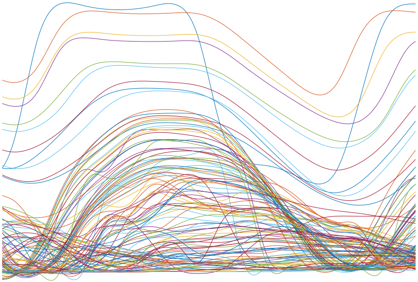
chlorine concentration levels
120 junctions, 180 time slots



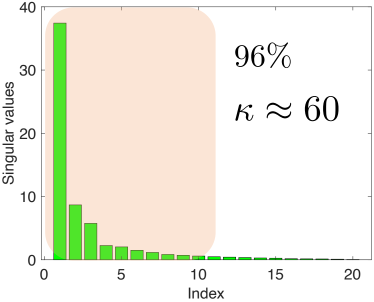
rank-5 approximation

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

Condition number can be large



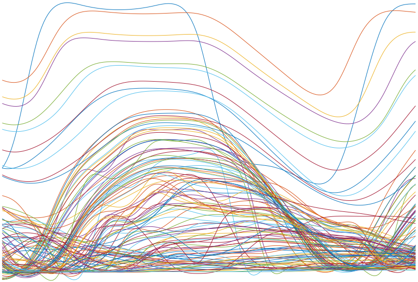
chlorine concentration levels
120 junctions, 180 time slots



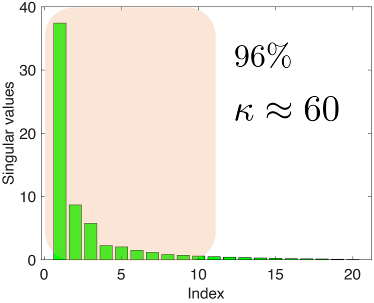
rank-10 approximation

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

Condition number can be large



chlorine concentration levels
120 junctions, 180 time slots



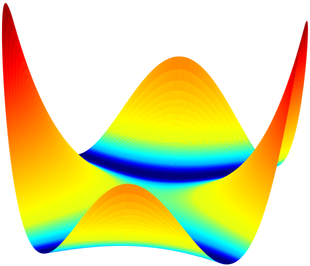
rank-10 approximation

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

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

A recipe: scaled gradient descent (ScaledGD)

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



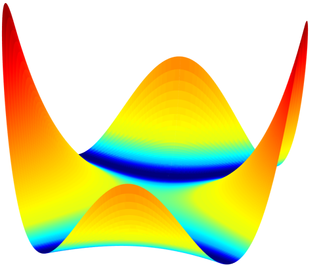
- **Spectral initialization:** find an initial point in the "basin of attraction"
- **Scaled gradient iterations:** for $t = 0, 1, \ldots,$

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

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

A recipe: scaled gradient descent (ScaledGD)

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



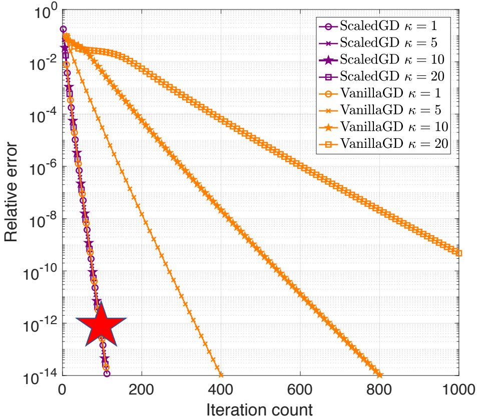
- **Spectral initialization:** find an initial point in the "basin of attraction"
- **Scaled gradient iterations:** for $t = 0, 1, \ldots,$

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

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

ScaledGD is a *preconditioned* gradient method
without balancing regularization

ScaledGD for low-rank matrix completion



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

A closer look at ScaledGD

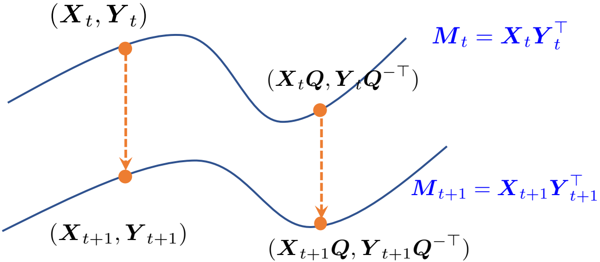
Connection to quasi-Newton method :

Define $\boldsymbol{F}_t = [\boldsymbol{X}_t^\top, \boldsymbol{Y}_t^\top]^\top \in \mathbb{R}^{(n_1+n_2)\times r}$. One can write update rule as

$$
\begin{aligned}
&\mathrm{vec}(\boldsymbol{F}_{t+1})\\
&= \mathrm{vec}(\boldsymbol{F}_t) - \eta \underbrace{\begin{bmatrix} (\boldsymbol{R}_t^\top \boldsymbol{R}_t)^{-1} \otimes \boldsymbol{I}_{n_1} & \boldsymbol{0} \\ \boldsymbol{0} & (\boldsymbol{L}_t^\top \boldsymbol{L}_t)^{-1} \otimes \boldsymbol{I}_{n_2} \end{bmatrix}}_{=:\boldsymbol{H}_t^{-1}} \mathrm{vec}(\nabla_{\boldsymbol{F}} \mathcal{L}(\boldsymbol{F}_t))
\end{aligned}
$$

A closer look at ScaledGD

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



— not true for GD

Theoretical guarantees of ScaledGD

Theorem 2 (Tong, Ma and Chi, JMLR 2021)

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 2 (Tong, Ma and Chi, JMLR 2021)

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

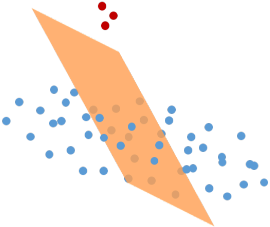
$$\|\boldsymbol{X}_t \boldsymbol{Y}_t^\top - \boldsymbol{M}\|_{\mathrm{F}} \lesssim \varepsilon \cdot \sigma_{\min}(\boldsymbol{M})$$

- ***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 with 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 computational saving at comparable sample complexities

What if we do not know the exact rank?

So far we have assumed the exact rank is given.... what if we do not know the exact rank?

What if we do not know the exact rank?

So far we have assumed the exact rank is given.... what if we do not know the exact rank?

Misspecification by overparameterization:

$$\boldsymbol{M} = \boldsymbol{X}\boldsymbol{X}^\top, \qquad \boldsymbol{X} \in \mathbb{R}^{n \times \tilde{r}}, \qquad \tilde{r} > r$$

What if we do not know the exact rank?

So far we have assumed the exact rank is given.... what if we do not know the exact rank?

Misspecification by overparameterization:

$$\boldsymbol{M} = \boldsymbol{X}\boldsymbol{X}^\top, \qquad \boldsymbol{X} \in \mathbb{R}^{n\times \tilde{r}}, \qquad \tilde{r} > r$$

ScaledGD:

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

analysis break down and might be unstable...

What if we do not know the exact rank?

So far we have assumed the exact rank is given.... what if we do not know the exact rank?

Misspecification by overparameterization:

$$\boldsymbol{M} = \boldsymbol{X}\boldsymbol{X}^\top, \qquad \boldsymbol{X} \in \mathbb{R}^{n\times\tilde{r}}, \qquad \tilde{r} > r$$

ScaledGD(λ):

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

add regularization to stablize the preconditioner

Theoretical guarantees

Theorem 3 (Xu, Shen, Chi, Ma, ICML 2023)

For low-rank matrix sensing with i.i.d. Gaussian design, overparameterized ScaledGD(λ) with $\lambda \asymp \sigma_{\min}(\boldsymbol{M})$, $\eta \asymp 1$, and a sufficiently small random initialization achieves

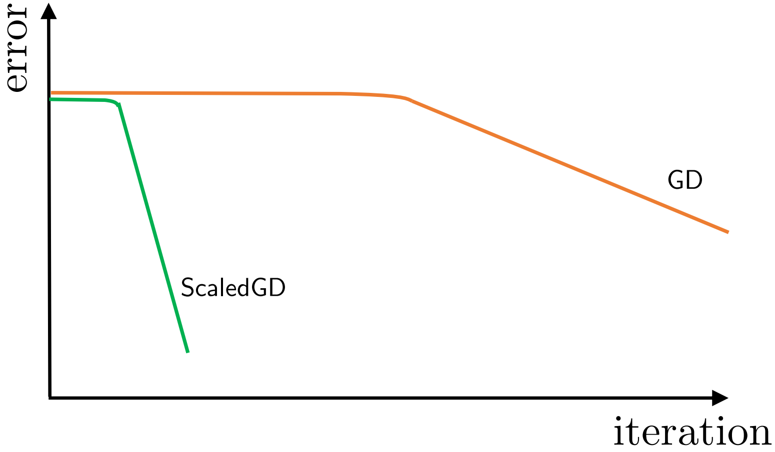
$$\|\boldsymbol{X}_t \boldsymbol{X}_t^\top - \boldsymbol{M}\|_{\mathrm{F}} \lesssim \varepsilon \cdot \sigma_{\min}(\boldsymbol{M})$$

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

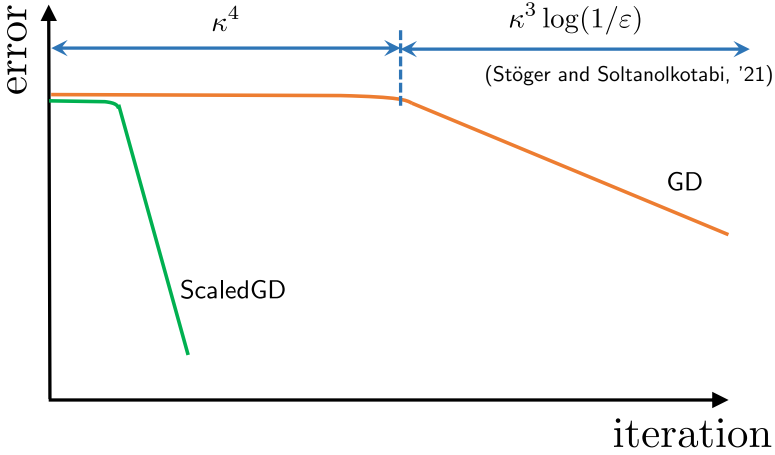
$$m \gtrsim n r^2 \mathsf{poly}(\kappa)$$

- Our analysis also enables exact convergence under random initialization with correct rank specification

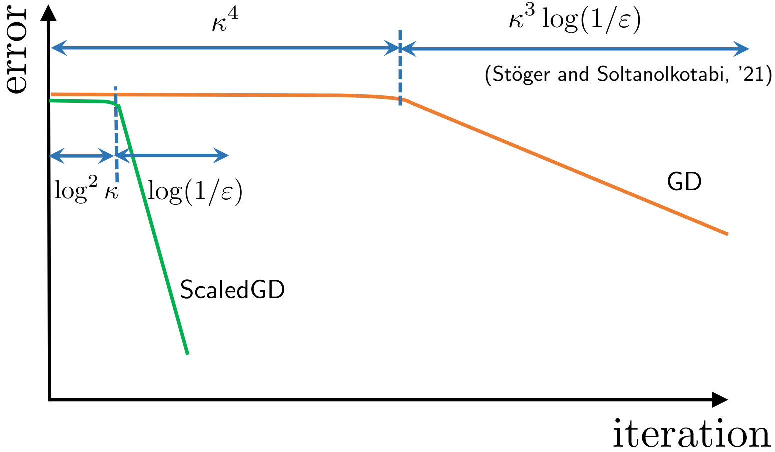
Comparison with overparameterized GD



Comparison with overparameterized GD



Comparison with overparameterized GD



ScaledGD picks up the signal component much faster than GD even from small random initialization

Comparisons with prior art

Comparison with Zhang, Fattahi, and Zhang '21

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

where $\lambda_t = \|\mathcal{A}(\boldsymbol{X}_t \boldsymbol{X}_t^\top - \boldsymbol{M})\|$

- Local analysis: require spectral initialization
- Large sample complexity: sample complexity is $n\tilde{r}^2 \,\mathrm{poly}(\kappa)$, depending on the overparameterized rank $\tilde{r}$ rather than the true rank r

Extension to noisy case

Consider the noisy setting

$$y_i = \langle A_i, \boldsymbol{M} \rangle + \xi_i, \quad \text{where} \quad \xi_i \sim \mathcal{N}(0, \sigma^2)$$

Theorem 4 (Xu, Shen, Chi, Ma, '23)

For low-rank matrix sensing with i.i.d. Gaussian design, overparameterized ScaledGD(λ) with the same configuration as before achieves

$$\|\boldsymbol{X}_t \boldsymbol{X}_t^\top - \boldsymbol{M}\|_{\mathrm{F}} \lesssim \kappa^2 \sigma \sqrt{nr}$$

ScaledGD(λ) is nearly optimal

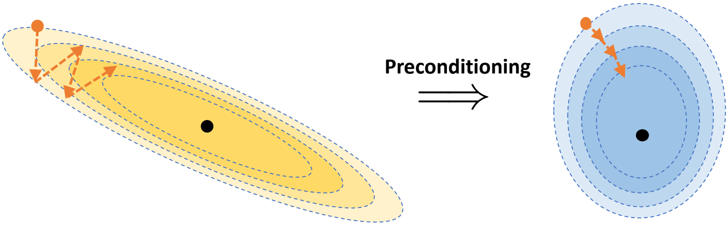
ScaledGD(λ) achieves

$$\|\boldsymbol{X}_t \boldsymbol{X}_t^\top - \boldsymbol{M}\|_{\mathrm{F}} \lesssim \kappa^2 \sigma \sqrt{nr}$$

- ScaledGD(λ) is minimax optimal (up to κ^2) for recovering rank-r matrices, cf. Candès and Plan '09
- Both the rate and sample size requirement improve over prior art (e.g., Zhuo et al., '21, Zhang et al., '23) as ours depend on true rank r

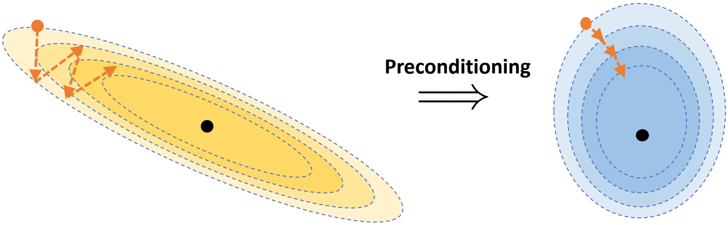
Concluding remarks

Preconditioning helps!



Preconditioning can dramatically increase the computational efficiency of vanilla gradient methods without hurting statistical efficiency

Preconditioning helps!



Preconditioning can dramatically increase the computational efficiency of vanilla gradient methods without hurting statistical efficiency

Future directions:

- streaming/stochastic variants of ScaledGD
- generalizing the idea of ScaledGD to other learning problems

Papers:

"The power of preconditioning in overparameterized low-rank matrix sensing," X. Xu, Y. Shen, Y. Chi, and C. Ma, ICML 2023

"Accelerating ill-conditioned low-rank matrix estimation via scaled gradient descent," T. Tong, C. Ma, and Y. Chi, JMLR 2021