Cyclic Block Coordinate Methods on a Finer Scale: Tighter Bounds and New Methods

Jelena Diakonikolas (UW-Madison)

Jason's Optimization Seminar

Outline

- Recap of coordinate methods (cyclic vs randomized)
- A new cyclic method for variational inequalities
- Generalizations
- A fun problem I am looking at (if enough time!)

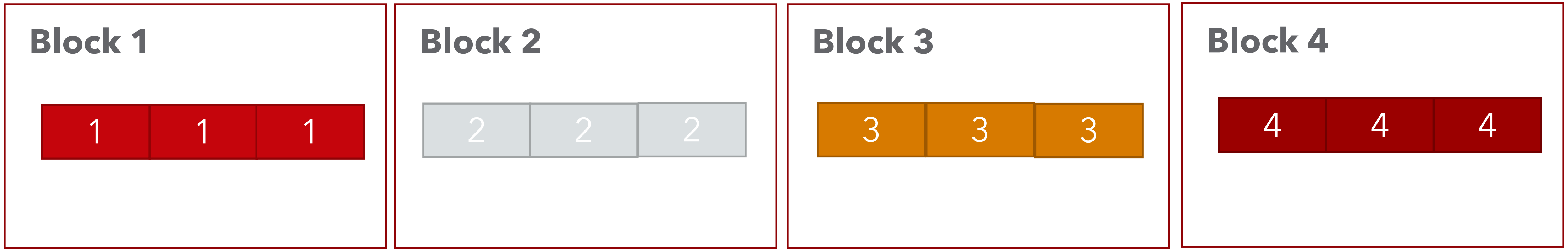
A Quick Refresher on (Block) Coordinate Methods

- Fix a partition of the vector of variables into m blocks:

1	2	1	3	3	1	2	4	4	2	3	4

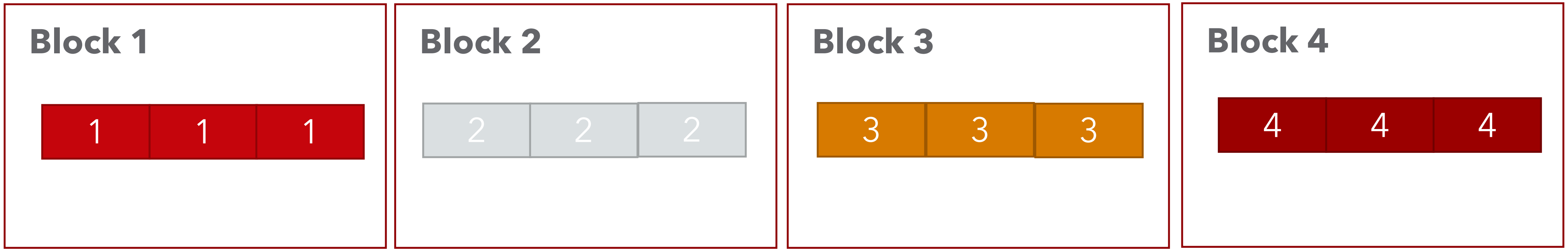
A Quick Refresher on (Block) Coordinate Methods

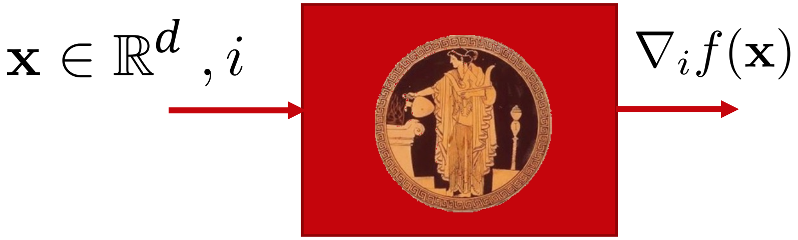
- Fix a partition of the vector of variables into m blocks:



A Quick Refresher on (Block) Coordinate Methods

- Fix a partition of the vector of variables into m blocks:

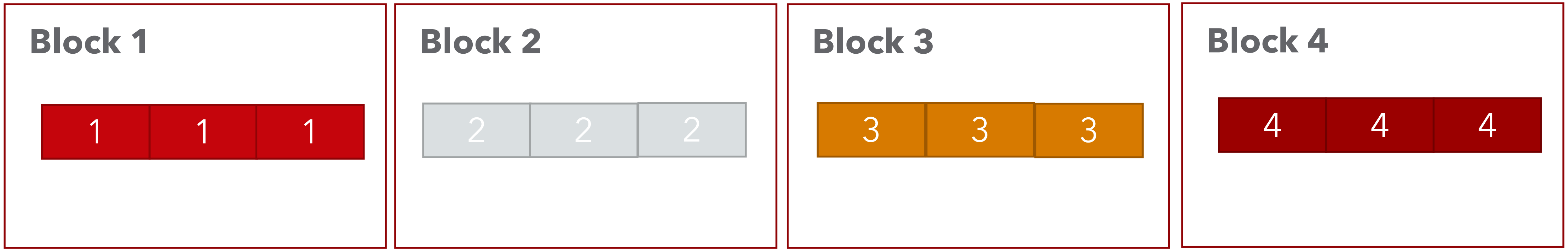




first-order oracle

A Quick Refresher on (Block) Coordinate Methods

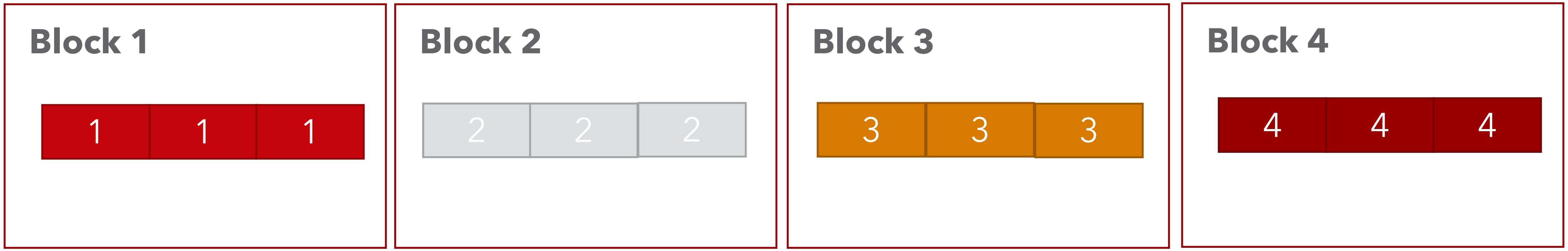
- Fix a partition of the vector of variables into m blocks:



- Types of methods/orderings of updates:
  - **cyclic**: fix an order of blocks, go through all of them in a cycle;
  - **randomized**: pick blocks randomly, sample with replacement;
  - **greedy**: pick the block that leads to the largest progress

A Quick Refresher on (Block) Coordinate Methods

- Fix a partition of the vector of variables into m blocks:



- Types of methods/orderings of updates:
  - **cyclic**: fix an order of blocks, go through all of them in a cycle;
  - **randomized**: pick blocks randomly, sample with replacement;
  - **greedy**: pick the block that leads to the largest progress

Theory vs Practice?

- Randomized methods [Strohmer & Vershynin’08], [Nesterov’12]:
  - almost always faster than full gradient methods, assuming the problem is (block) coordinate friendly [Nesterov’12] + a lot of follow-up work;
  - generally the best theoretical guarantees among block coordinate methods;
  - key property: can relate the partial gradient to the full one by taking the expectation, helps much of the analysis carry over from full gradient methods
- Cyclic methods [Kaczmarz’37], [Ortega & Rheinboldt’70]:
  - often preferred in practice over randomized methods (e.g., in GLMNet, SparseNet);
  - much more challenging to analyze; hard to relate partial gradients to full ones;
  - complexity guarantees generally worse than even for full gradient methods and smooth cvx opt, by dimension-dependent factors [Beck-Tetruashvili’13]; this is tight for cyclic gradient descent-type method [Sun-Ye’19]

State of the Art for Cyclic Methods (Prior to This Work)

- Essentially **no non-asymptotic results** for min-max opt/variational inequalities
  - Exception: [Chow-Wu-Yin'17], but requires cocoercivity and the rate is $1/\sqrt{k}$
- For (non-accelerated) smooth convex optimization, number of full gradient queries (assuming coordinate-friendly) of the order $O\left(\frac{mLD^2}{\epsilon}\right)$, at best (**worse by a factor m =# of blocks than gradient descent; $m = d$ for coordinate descent**)

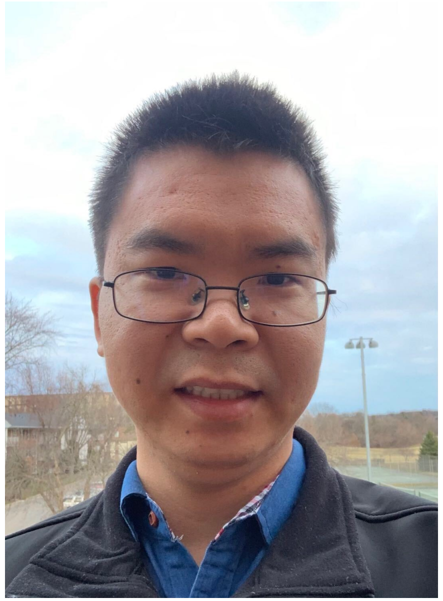
The only exception are convex quadratic problems [Gürbüzbalaban et al., 2017], [Lee & Wright, 2019]

- *All analyses based on **relating the partial gradient to the full one**

*as far as I can tell, please correct me if wrong

Cyclic Coordinate Dual Averaging with Extrapolation (CODER)

Setup and Results



Chaobing Song (Huawei)

Joint Work With:

C. Song, J. Diakonikolas, *"Cyclic Coordinate Dual Averaging with Extrapolation," SIAM Journal on Optimization, vol. 33, no. 4, pp. 2935-2961, 2023.*

Problem Setup

(GMVI) Find $x^* \in \mathbb{R}^d$ s.t. $\forall x, \langle F(x), x - x^*\rangle + g(x) - g(x^*) \geq 0$

($\mathrm{P_{CO}}$) $\min_{x \in \mathbb{R}^d} f(x) + g(x)$

($\mathrm{P_{MM}}$) $\min_x \max_y \Phi(x, y)$

- Assumptions:
  - There is a fixed partition of coordinates into m blocks;
  - $F: \mathbb{R}^d \to \mathbb{R}^d$ is
    - "(block) coordinate-friendly" according to that partition;
    - monotone: $\forall x, y$: $\langle F(x) - F(y), x - y\rangle \geq 0$;
    - Lipschitz: $\exists L < \infty$ s.t. $\forall x, y$: $\|F(x) - F(y)\| \leq L\|x - y\|$
  - $g: \mathbb{R}^d \to \mathbb{R}$ is
    - block-separable over the given partition: $g(x) = \sum_{j=1}^m g^j(x^j)$;
    - $\mathrm{prox}_{\tau g^j}(x^j) = \arg\min_{y \in \mathbb{R}^{d_j}} \left\{ g^j(y) + \frac{1}{2\tau}\left\|y - x^j\right\|^2 \right\}$ is easily computable, $\forall j \in \{1, \dots, m\}$;
    - possibly strongly convex, with modulus $\gamma \geq 0$ and lower semicontinuous

I'll focus on the cyclic coordinate case ($m = d$), for simplicity

Coordinate Lipschitz Assumption?

- What is standard in convex optimization:

$$\left|\nabla^j f(x) - \nabla^j f\big(x + \underbrace{h}_{\text{scalar}}\ \overbrace{e_j}^{j^{\text{th}}\text{ standard basis vector}}\big)\right| \le L_j |h|, \qquad \forall x \in \mathbb{R}^d, h \in \mathbb{R}$$

- For VIs unclear how to make useful. Take bilinear games $\min_x \max_y x^T A y$. Then

$$F\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = \begin{bmatrix} 0 & A \\ -A^T & 0 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} Ay \\ -A^T x \end{bmatrix},$$

so $F^j\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = A_{j:}y$ for j in the block belonging to the x-player. In particular,

$$F^j\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) - F^j\left(\begin{bmatrix} x + h\, e_j \\ y \end{bmatrix}\right) = 0$$

A Different Coordinate Lipschitz Condition?

- Going back to the bilinear case, for any $\begin{bmatrix} x \\ y \end{bmatrix}, \begin{bmatrix} x' \\ y' \end{bmatrix}$ and any j in the x-part,

$$F^j\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) - F^j\left(\begin{bmatrix} x' \\ y' \end{bmatrix}\right) = A_{j:}(y - y')$$

- From there, we can conclude:

quadratic form

$$\left|F^j\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) - F^j\left(\begin{bmatrix} x' \\ y' \end{bmatrix}\right)\right|^2 = \boxed{\begin{bmatrix} x - x' \\ y - y' \end{bmatrix}^T \boxed{\begin{bmatrix} 0 & 0 \\ 0 & A_{j:}A_{j:}^T \end{bmatrix}} \begin{bmatrix} x - x' \\ y - y' \end{bmatrix}}.$$

symmetric
PSD matrix

- **Idea**: generalize to other (possibly nonlinear) operators F

New Lipschitz Condition

There exist symmetric positive semidefinite matrices $Q^1, Q^2, \ldots, Q^d$ such that for any $z, z' \in \mathbb{R}^d$,

$$\left|F^j(z) - F^j(z')\right|^2 \le (z - z')^T Q^j (z - z').$$

- Can be trivially satisfied with $Q^j = LI$, but can generally choose better
- Define:

$$\hat{Q}^j = \begin{array}{cc} \overbrace{}^{1,2,\ldots,j-1} & \\ \begin{bmatrix} 0 & 0 \\ 0 & Q^j \end{bmatrix} & \begin{array}{l} \} \; 1,2,\ldots,j-1 \\ \end{array} \end{array}$$

- Summary Lipschitz constant:

$$\hat{L} = \sqrt{\left\| \sum_{j=1}^{d} \hat{Q}^j \right\|}$$

Main Result

A method (CODER) that for any $\epsilon > 0$ outputs a solution with primal-dual gap at most ϵ, in

$$O\left(\min\left\{\frac{\hat{L}D^2}{\epsilon}, \frac{\hat{L}}{\gamma}\log\left(\frac{\hat{L}D}{\epsilon}\right)\right\}\right)$$

iterations.

- This is the same as for full-vector update methods, but with L replaced by $\hat{L}$

How Large is $\hat{L}$?

- Worst case:

$$\hat{L} = \sqrt{\left\|\sum_{j=1}^{d} \hat{Q}^j\right\|} \underset{\text{triangle inequality}}{\leq} \sqrt{\sum_{j=1}^{d} \|\hat{Q}^j\|} \underset{\hat{Q}^j \preccurlyeq Q^j}{\leq} \sqrt{\sum_{j=1}^{d} \|Q^j\|} \underset{Q^j \preccurlyeq LI}{\leq} L\sqrt{d}$$

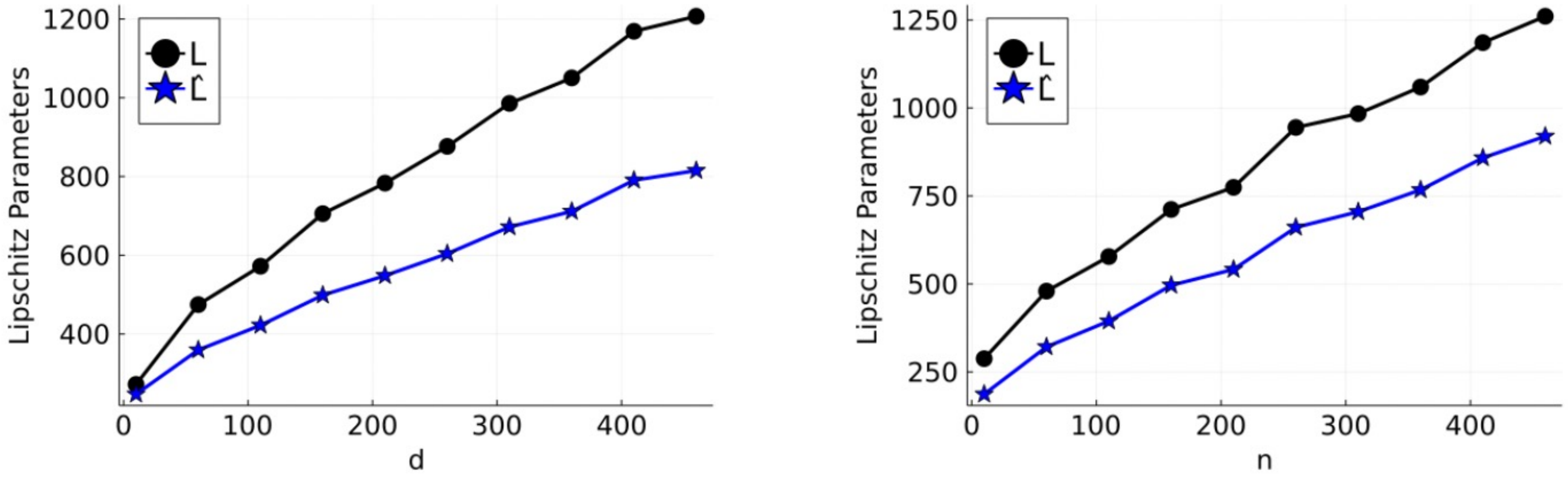
- Even for the special case of smooth convex optimization, the resulting bound is better by a factor $\sqrt{d}$ than what was known for any (unaccelerated) cyclic coordinate method.
- For variational inequalities, the obtained complexity result is state-of-the-art, even compared to randomized methods [Kotsalis et al., 2022] (it is actually better by a factor $\sqrt{d}$ in the worst case).
- It is also the first cyclic method for general variational inequalities with monotone operators with provable convergence guarantees.

How Large is $\hat{L}$?

- In practice, $\hat{L}$ is no larger than L, usually smaller (constants below are for min-max SVM)
  - Real data sets:

Dataset	a9a	australia	madelon	colon	mnist
L	15389.6	340.5	1992.4	9.0	24410.5
$\hat{L}$	10358.8	238.1	1269.7	5.7	15236.1

  - Synthetic (standard Gaussian) data; fixed dataset size n or fixed dimension d, respectively



Algorithm

Algorithm 3.1 Cyclic cOordinate Dual avEraging with extRapolation (CODER)

Input: $\boldsymbol{x}_{-1} = \boldsymbol{x}_0 \in \mathrm{dom}(g), \gamma \geq 0, \hat{L} > 0, m, \{\mathcal{S}^1, \ldots, \mathcal{S}^m\}$
Initialization: $\boldsymbol{p}_0 = \boldsymbol{F}(\boldsymbol{x}_0), \boldsymbol{z}_0 = \boldsymbol{0},\ a_0 = A_0 = 0$
for $k = 1$ to K **do**
    $a_k = \frac{1+\gamma A_{k-1}}{2\hat{L}}, A_k = A_{k-1} + a_k$
    for $j = 1$ to m **do**
        $\boldsymbol{p}_k^j = \boldsymbol{F}^j(\boldsymbol{x}_k^1, \ldots, \boldsymbol{x}_k^{j-1}, \boldsymbol{x}_{k-1}^j, \ldots \boldsymbol{x}_{k-1}^m)$ ← partial "gradient" at intermediate point
        $\boldsymbol{q}_k^j = \boldsymbol{p}_k^j + \frac{a_{k-1}}{a_k}(\boldsymbol{F}^j(\boldsymbol{x}_{k-1}) - \boldsymbol{p}_{k-1}^j)$ ← partial "gradient" extrapolation
        $\boldsymbol{z}_k^j = \boldsymbol{z}_{k-1}^j + a_k \boldsymbol{q}_k^j$
        $\boldsymbol{x}_k^j = \mathrm{prox}_{A_k g^j}(\boldsymbol{x}_0^j - \boldsymbol{z}_k^j)$
        } dual averaging step with extrapolated gradient
    end for
end for
return $\tilde{\boldsymbol{x}}_K = \frac{1}{A_K}\sum_{k=1}^{K} a_k \boldsymbol{x}_k,\ \boldsymbol{x}_K$

Similar gradient extrapolation in [Hamedani & Aybat., 2018] and [Kotsalis et al., 2022]

How the Analysis Works

$$\text{(GMVI) Find } x^* \in \mathbb{R}^d \text{ s.t. } \forall x, \langle F(x), x - x^* \rangle + g(x) - g(x^*) \geq 0$$

- We can only solve this approximately to error $\epsilon > 0$, so equivalently want to

$$\text{find } x^*_\epsilon \in \mathbb{R}^d \text{ s.t. } \forall x, \langle F(x), x - x^*_\epsilon \rangle + g(x) - g(x^*_\epsilon) \geq -\epsilon$$

  which is the same as find $x^*_\epsilon \in \mathbb{R}^d$ s.t. $\forall x$, $\boxed{\mathrm{Gap}(x, x^*_\epsilon) \coloneqq \langle F(x), x^*_\epsilon - x \rangle - g(x) + g(x^*_\epsilon) \leq \epsilon}$

- So we may try to bound $\mathrm{Gap}(x, x_k)$ for iterations $k = 1, 2, \ldots$
- The key step:

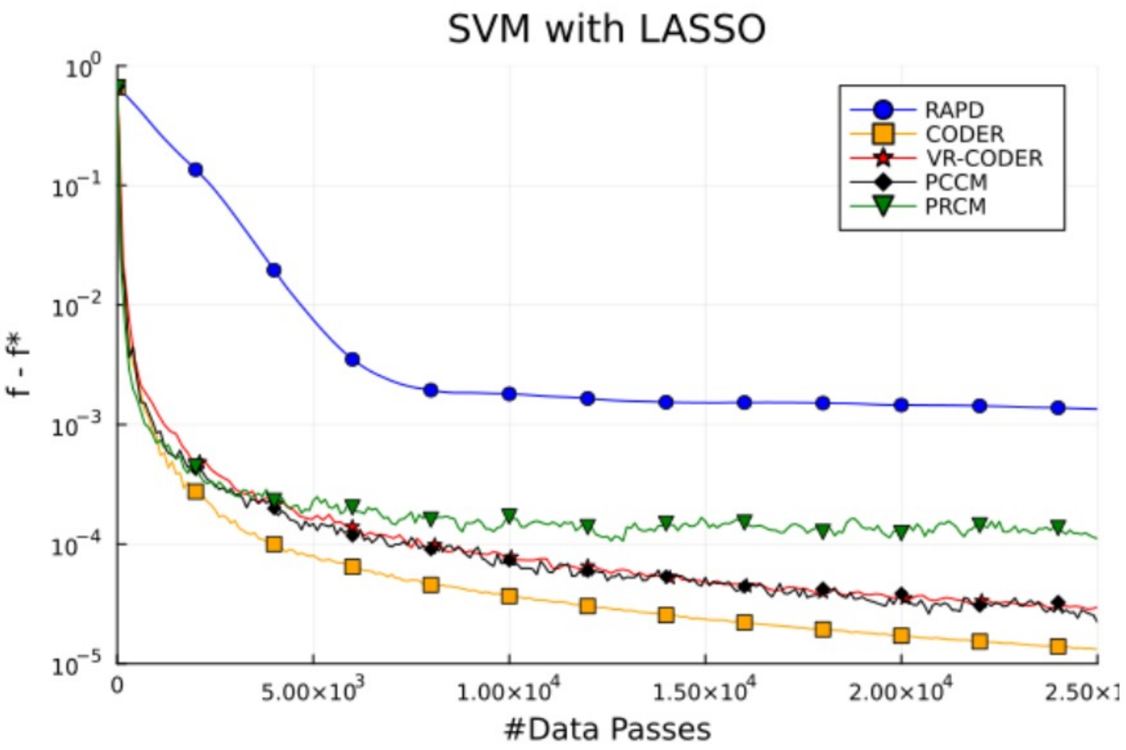
$$\begin{aligned} \mathrm{Gap}(x, x_k) &= \langle F(x), x_k - x \rangle - g(x) + g(x_k) \\ &\leq \langle F(x_k), x_k - x \rangle - g(x) + g(x_k) \\ &= \boxed{\langle q_k, x_k - x \rangle - g(x) + g(x_k) - \frac{1}{2}\|x - x_k\|_2^2} + \frac{1}{2}\|x - x_k\|_2^2 \\ &\quad + \boxed{\langle F(x_k) - q_k, x_k - x \rangle} \end{aligned}$$

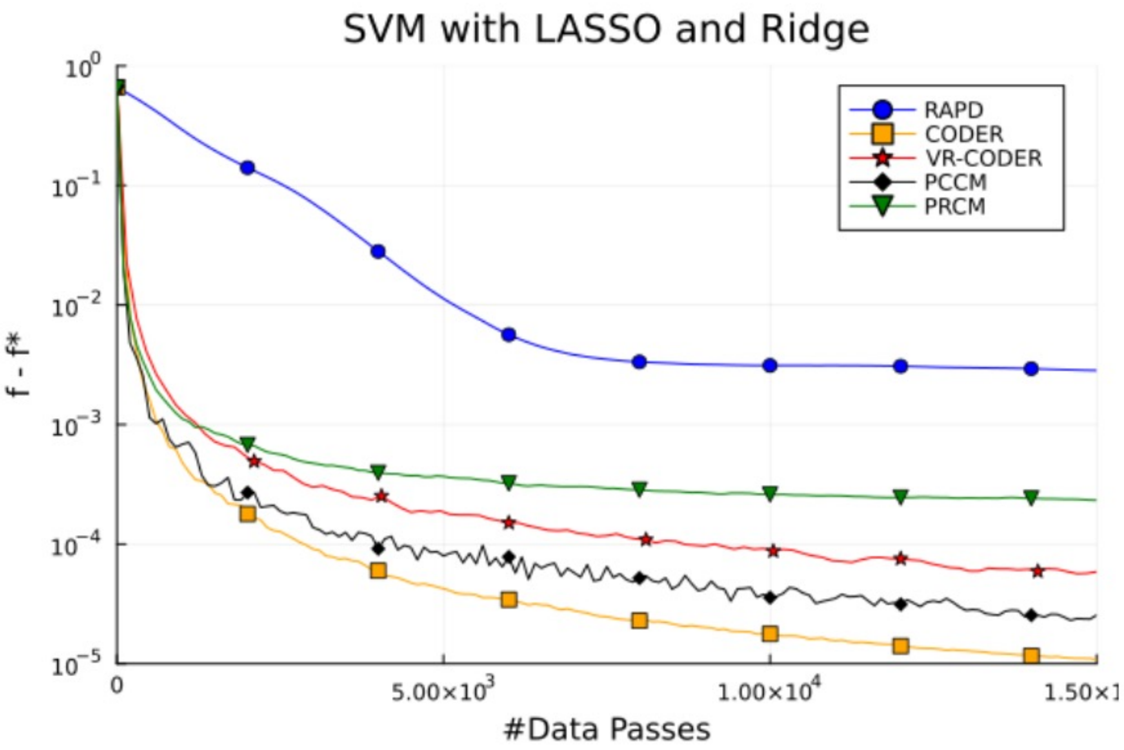
this part defines the step, by taking a max

the art of choosing q_k is for controlling this term

Numerical Experiments (Illustration)

SVM with LASSO or ridge, on a1a LibSVM dataset ($d = 123, n = 1605$)





Further Developments

Two Examples: Bilinear Extensive Form Games and Shuffled SGD

Q: Are cyclic methods always slower than full vector update methods in the worst case?

A: No, and they may have advantages. A specific example (the first of its kind!) to follow.

Joint Work With:

Darshan Chakrabarti
(Columbia University)

Christian Kroer
(Columbia University)

D. Chakrabarti, J. Diakonikolas, C. Kroer "*Block Coordinate Methods and Restarting for Solving Extensive Form Games," in Proc. NeurIPS 2023*. ($\alpha\beta$ ordering of authors)

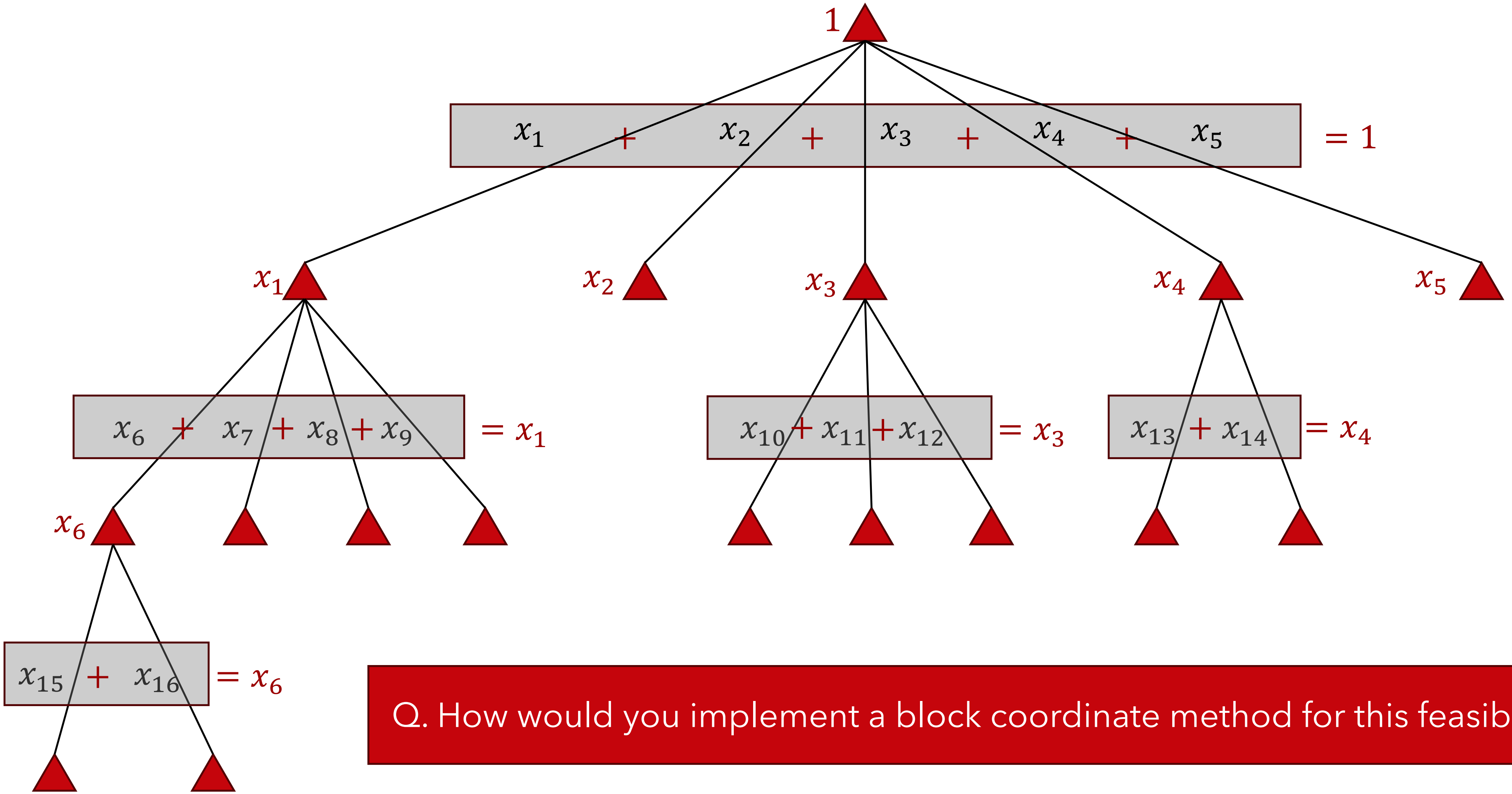
Problem: Bilinear Extensive Form Games (EFGs)

- A general class of game-theoretic models that capture both simultaneous and sequential moved, private/imperfect information, and stochasticity
- In optimization language, we have a bilinear min-max problem:

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \boldsymbol{y}^\top \boldsymbol{M} \boldsymbol{x}$$

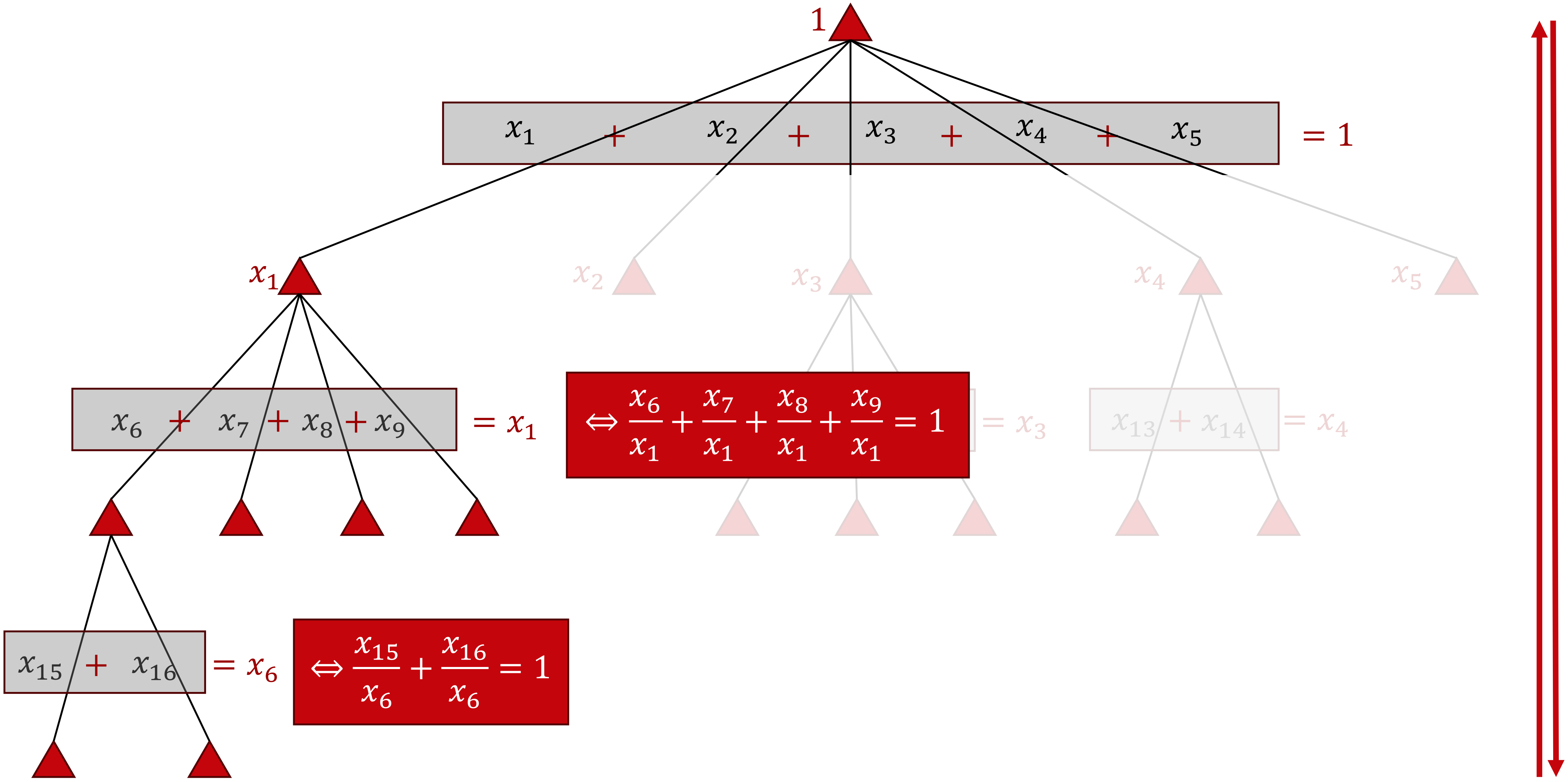
sequence form polytopes: treeplexes

Treeplex?



Q. How would you implement a block coordinate method for this feasible set?

Why Cyclic Updates are OK



Main Result

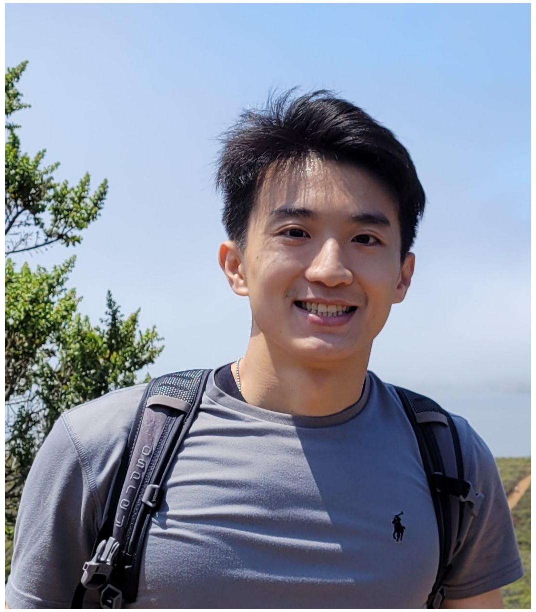
A method (ECyclicPDA) that for any $\epsilon > 0$ outputs a solution with primal-dual gap at most ϵ with iteration complexity that is **no worse** than the iteration complexity of full-vector update methods like Mirror-Prox.

This is the first example (that I know of) of a cyclic method with no scaling with the number of blocks in the worst case.

Q: Do these ideas extend beyond basic cyclic coordinate methods?

A: Yes. They extend to the incremental gradient method and shuffled SGD.

Xufeng Cai
(UW-Madison)



Eric (Cheuk-Yin) Lin
(UW-Madison)

Joint Work With:

X. Cai, C-Y. Lin, J. Diakonikolas, "Empirical Risk Minimization with Shuffled SGD: A Primal-Dual Perspective and Improved Bounds," arXiv preprint, arXiv:2306.12498, 2023.

Problem Setup

Empirical Risk Minimization problems over data $\{\boldsymbol{a}_1, \boldsymbol{a}_2, \dots, \boldsymbol{a}_n\}$

$$\min_{\boldsymbol{x}} \frac{1}{n} \sum_{i=1}^{n} \ell_i(\boldsymbol{a}_i^\top \boldsymbol{x})$$

are (in practice) usually solved by incremental/shuffled SGD methods, which make updates

$$\boldsymbol{x}_{k,i+1} = \boldsymbol{x}_{k,i} - \eta \nabla \ell_i(\boldsymbol{a}_i^\top \boldsymbol{x}_{k,i})$$

going through all the data vectors in a cyclic manner, possibly permuting the order at the beginning of a cycle.

Convergence guarantees: not understood until recently [Gürbüzbalaban et al., 2021], [Shamir, 2016], [Haochen & Sra, 2019], [Nagaraj et al., 2019], [Rajput et al., 2020], [Ahn et al., 2020], [Mishchenko et al., 2020], [Nguyen et al., 2021], [Cha et al., 2023]

Main Insight

- We can view incremental gradient/shuffled SGD as a primal-dual method with cyclic updates on the dual side

Algorithm 1 Shuffled SGD (Primal-Dual View)

Input: Initial point $\boldsymbol{x}_0 \in \mathbb{R}^d$, batch size $b > 0$, step size $\{\eta_k\} > 0$, number of epochs $K > 0$
for $k = 1$ to K **do**
  Generate any permutation $\pi^{(k)}$ of $[n]$ (either deterministic or random)
  $\boldsymbol{x}_{k-1,1} = \boldsymbol{x}_{k-1}$
  for $i = 1$ to m **do**
    $\boldsymbol{y}_k^{(i)} = \arg\max_{\boldsymbol{y} \in \mathbb{R}^b} \left\{ \boldsymbol{y}^\top \boldsymbol{A}_k^{(i)} \boldsymbol{x}_{k-1,i} - \sum_{j=1}^{b} \ell^*_{\pi^{(k)}_{b(i-1)+j}}(\boldsymbol{y}^j) \right\}$
    $\boldsymbol{x}_{k-1,i+1} = \arg\max_{\boldsymbol{x} \in \mathbb{R}^d} \left\{ \boldsymbol{y}_k^{(i)\top} \boldsymbol{A}_k^{(i)} \boldsymbol{x} + \frac{b}{2\eta_k} \|\boldsymbol{x} - \boldsymbol{x}_{k-1,i}\|^2 \right\}$
  end for
  $\boldsymbol{x}_k = \boldsymbol{x}_{k-1,m+1}$, $\boldsymbol{y}_k = (\boldsymbol{y}_k^{(1)}, \boldsymbol{y}_k^{(2)}, \ldots, \boldsymbol{y}_k^{(m)})^\top$
end for
Return: $\hat{\boldsymbol{x}}_K = \sum_{k=1}^{K} \eta_k \boldsymbol{x}_k / \sum_{k=1}^{K} \eta_k$

Main Results

Tighter, data-dependent, convergence bounds for all standard variants of incremental gradient/shuffled SGD, for smooth convex loss functions. The obtained bounds are tighter by a factor that can be as large as $\sqrt{n}$, both in theory and in practice.

How Tighter in Practice?

Dataset	#Features (d)	#Datapoints (n)	$L/\hat{L}$	$\log_n L/\hat{L}$
A1A	123	1605	5.50	0.231
A9A	123	32561	5.49	0.164
BBBC005	361920	19201	18.3	0.295
BBBC010	361920	201	7.04	0.368
CIFAR10	3072	50000	10.0	0.213
DUKE	7129	44	38.0	0.962
E2006TRAIN	150360	16087	5.35	0.173
GISETTE	5000	6000	3.52	0.145
LEU	7129	38	32.8	0.960
MNIST	780	60000	19.1	0.268
NEWS20	1355191	19996	42.1	0.378
RCV1	47236	20242	111	0.475
REAL-SIM	20958	72309	194	0.471
SONAR	60	208	6.26	0.344
TMC2007	30438	21519	10.9	0.239

Other Extensions I Did Not Talk About

- Variance reduction for cyclic methods [Song, D, 2021], [Lin, Song, D, ICML 2023], [Cai, Song, Wright, D, ICML 2023]

- Acceleration in smooth convex optimization [Lin, Song, D, ICML 2023]

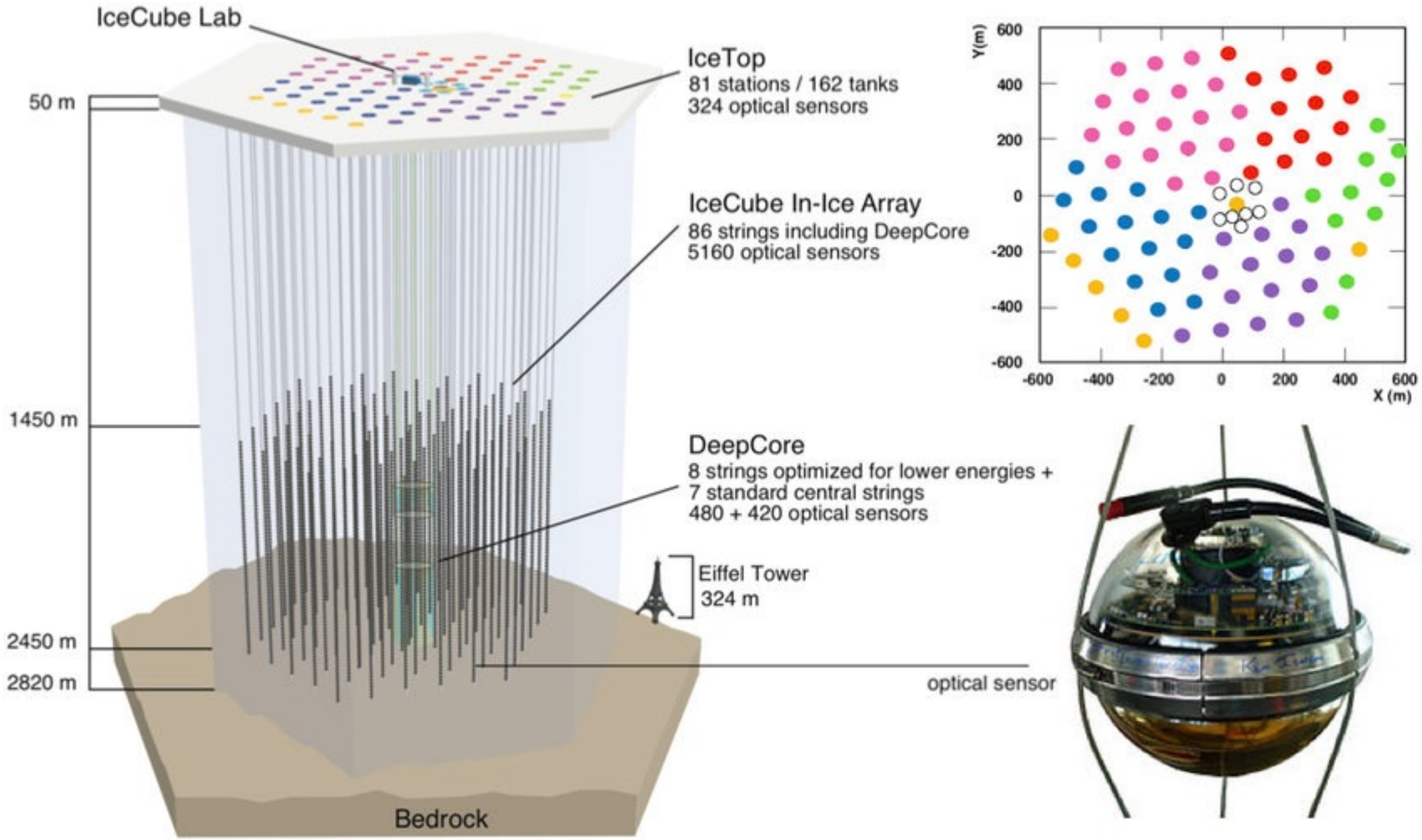
- Nonconvex optimization [Cai, Song, Wright, D, ICML 2023]

- Incremental gradient methods and continual learning [Cai, D, forthcoming]

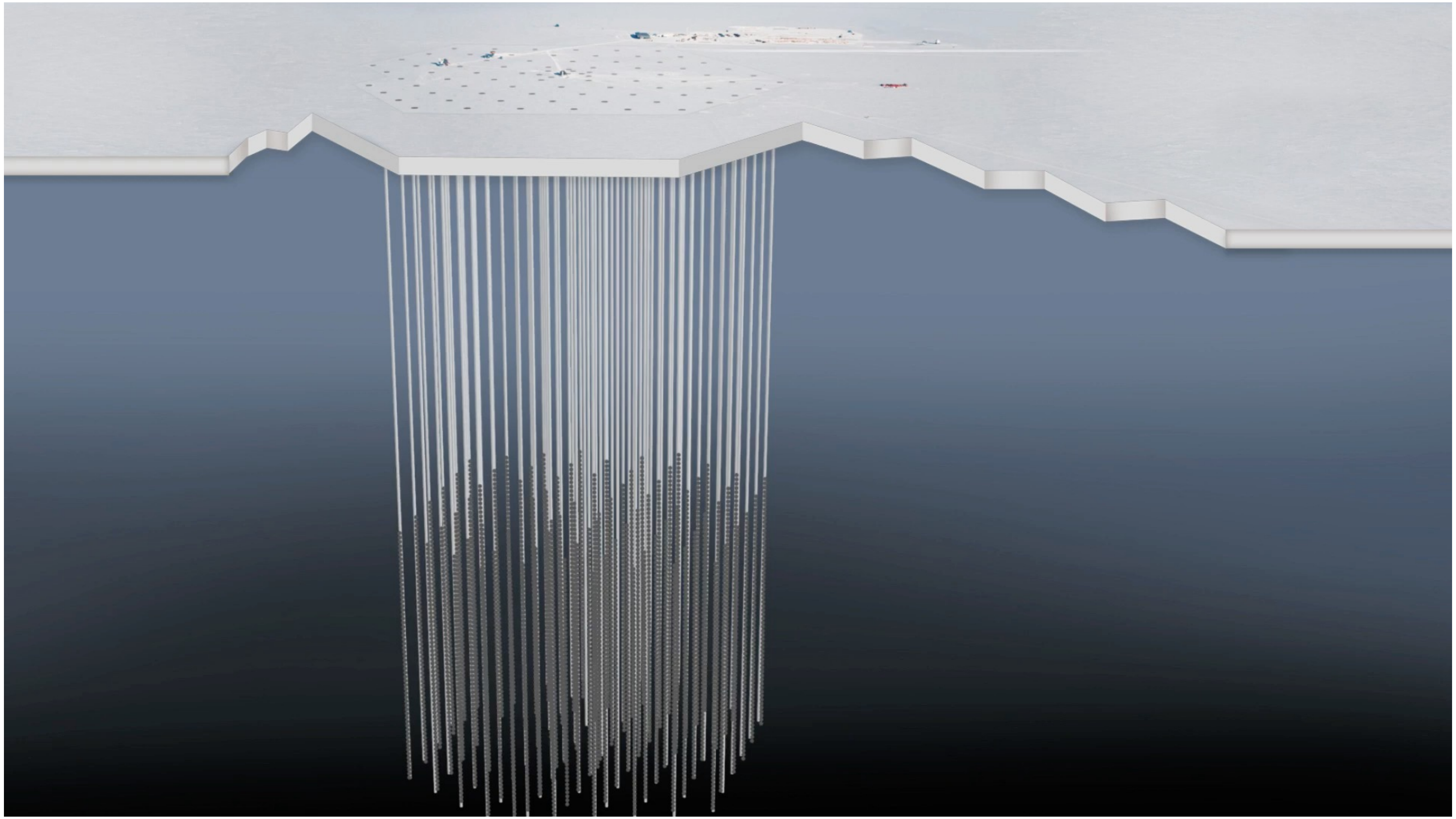
A Toy Project I am Excited About

and it is for a “real” application!

Ice Cube Neutrino Detector



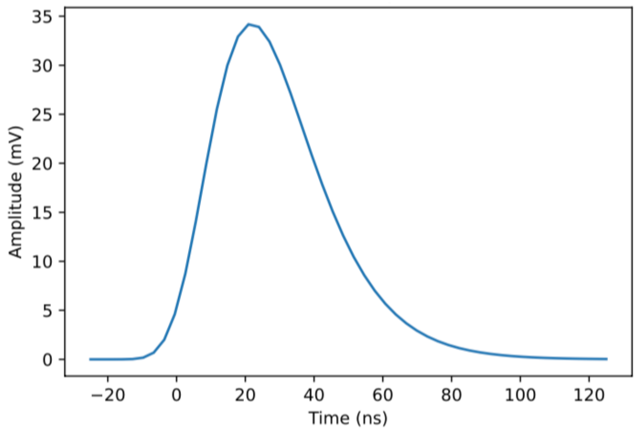
Ice Cube Neutrino Detector



Neutrino Detection & Non-negative Least Squares

Physics models for detected photon waveforms:

- $f(t) = A\left(e^{-\frac{t-x_0}{b_1}} + e^{\frac{t-x_0}{b_2}}\right)$
- $f(t) = A\left(Ce^{e^{-\frac{t-x_0}{b_1}}} + e^{\frac{t-x_0}{b_2}}\right)$



Problem can be formulated as a regularized non-negative least-square problem:

$$\boxed{\min_{\mathbf{x}\geq 0} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + r(\mathbf{x})}$$

where $\mathbf{A}$ is non-negative, highly sparse and structured, and the desired form of regularization $r(\mathbf{x})$ is unclear.

Summary

- We should still care about and study cyclic methods!
- We just need to be more careful about how we look at them
- What next?
  - For what classes of problems are cyclic methods particularly effective and why?
  - What kind of cyclic methods should we use in practice?

Questions?

jelena@cs.wisc.edu