Coordinate Update Algorithms for Optimization Problems in Machine Learning and Signal Processing

Wotao Yin (UCLA Math)

SAMSI OPT Opening Workshop – August 2016

Background

Large-sum decomposition

$$\underset{x\in\mathbb{R}^m}{\text{minimize}} \;\; r(x) + \frac{1}{N}\sum_{i=1}^{N} f_i(x)$$

- interested in large N
- nice structures: f_i's are smooth and r is proximable
- Stochastic approximation methods: SG, SAG, SAGA, SVRG, Finito

  pro: faster than batch methods

  con: update entire $x \in \mathbb{R}^m$; model is restricted

Coordinate descent decomposition

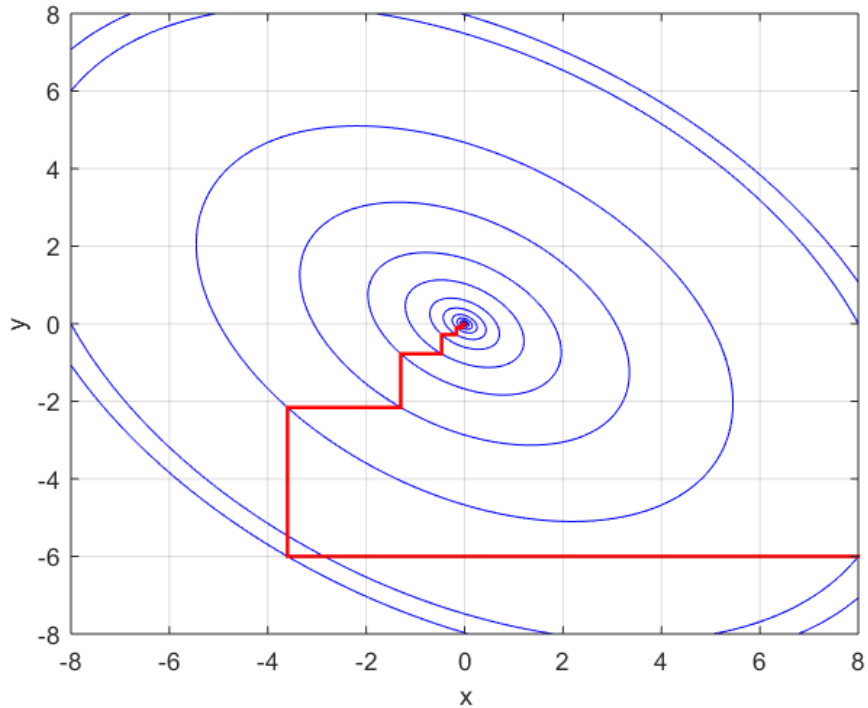
$$\underset{x \in \mathbb{R}^m}{\text{minimize}} \; f(x_1, \ldots, x_m) + \frac{1}{m} \sum_{i=1}^{m} r_i(x_i)$$

- f is smooth, r_i can be nonsmooth
- update variables in a (shuffled) cyclic, random, greedy, or parallel fashion

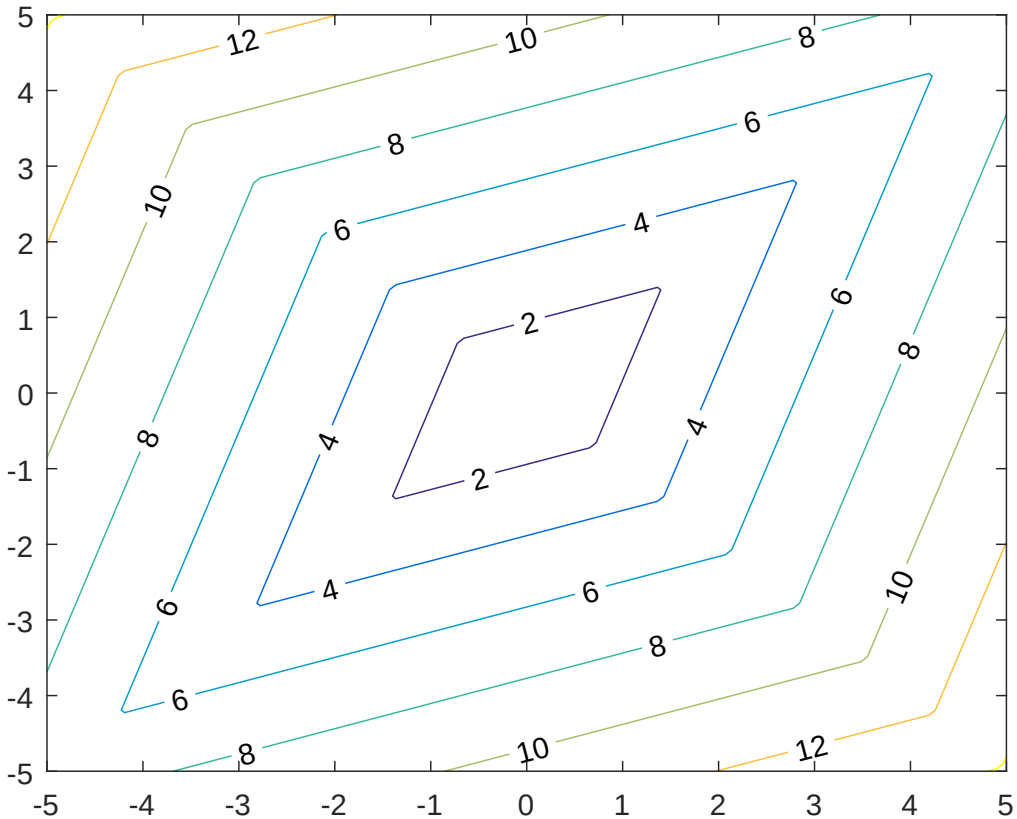
  pro: faster than the full-update method

  con: nonsmooth functions need to separable

CD illustration



CD fails on nonseparable nonsmooth functions



Rotated weighted ℓ_1-norm. CD fixed-point is not stationary!

- smooth $f(x_1, x_2)$

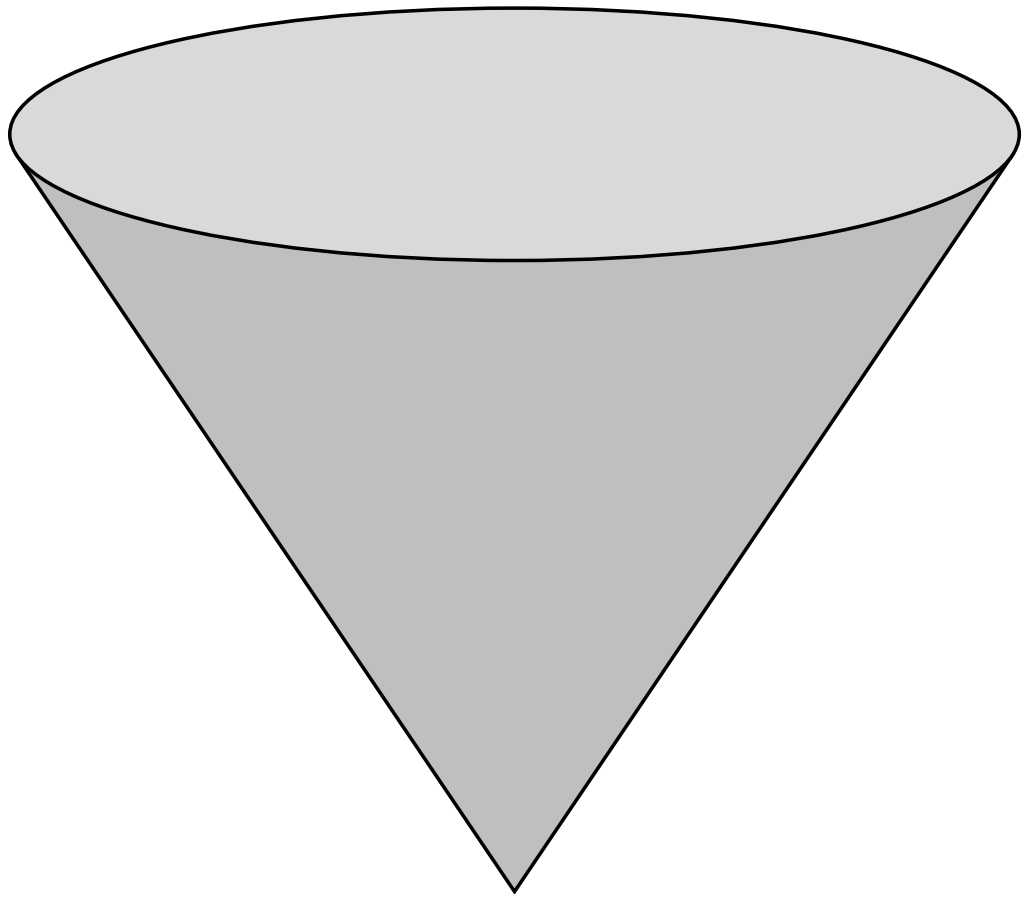
$$\begin{aligned} p_1 &= \nabla_1 f(x_1, x_2) \\ p_2 &= \nabla_2 f(x_1, x_2) \end{aligned} \quad \Longrightarrow \quad \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} = \nabla f(x_1, x_2)$$

- nonsmooth $r(x_1, x_2)$

$$\begin{aligned} q_1 &\in \partial_1 r(x_1, x_2) \\ q_2 &\in \partial_2 r(x_1, x_2) \end{aligned} \quad \not\Longrightarrow \quad \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \in \partial r(x_1, x_2)$$

(but it holds if r is separable. That is, $r(x_1, x_2) = r_1(x_1) + r_2(x_2)$.)

Cone constraint



What do we want to solve?

- total variation minimization:

$$\underset{x}{\text{minimize}} \ \mathrm{TV}(x) + \frac{1}{2}\|Ax - b\|^2$$

What do we want to solve?

- total variation minimization:

$$\underset{x}{\text{minimize}} \ \text{TV}(x) + \frac{1}{2}\|Ax - b\|^2$$

- optimization over a graph $G = (N, E)$

$$\underset{x}{\text{minimize}} \sum_{(i,j)\in E} f(x_i - x_j) + \sum_{i\in N} g_i(x_i)$$

What do we want to solve?

- total variation minimization:

$$\underset{x}{\text{minimize}} \ \text{TV}(x) + \frac{1}{2}\|Ax - b\|^2$$

- optimization over a graph $G = (N, E)$

$$\underset{x}{\text{minimize}} \sum_{(i,j)\in E} f(x_i - x_j) + \sum_{i\in N} g_i(x_i)$$

- consensus optimization (decentralized computing)

$$\underset{x_1,\ldots,x_m}{\text{minimize}} \ f_1(x_1) + f_2(x_2) + \cdots + f_m(x_m)$$

$$\text{subject to } x_1 = x_2 = \cdots = x_m$$

- second-order cone program

$$\begin{aligned} \underset{x_1,\ldots,x_m}{\text{minimize}} \;\; & c^T x + \frac{1}{2} x^T B x \\ \text{subject to} \;\; & A_1 x_1 + A_2 x_2 + \cdots + A_m x_m = b \\ & x_1 \in Q_1, x_2 \in Q_2 \ldots, x_m \in Q_m \end{aligned}$$

- second-order cone program

$$\begin{aligned} \underset{x_1,\ldots,x_m}{\text{minimize}} \;\; & c^T x + \frac{1}{2} x^T B x \\ \text{subject to} \;\; & A_1 x_1 + A_2 x_2 + \cdots + A_m x_m = b \\ & x_1 \in Q_1, x_2 \in Q_2 \ldots, x_m \in Q_m \end{aligned}$$

- extended monotropic program

$$\begin{aligned} \underset{x_1,\ldots,x_m}{\text{minimize}} \;\; & g_1(x_1) + g_2(x_2) + \cdots + g_m(x_m) + f(x) \\ \text{subject to} \;\; & A_1 x_1 \; + \; A_2 x_2 \; + \cdots + \; A_m x_m \; = b \end{aligned}$$

  where g_i are nonsmooth and extended-valued

Our approach

Overview

1. Re-formulate a problem into

$$x = T(x)$$

   (use operator splitting)

Overview

1. Re-formulate a problem into

$$x = T(x)$$

   (use operator splitting)

2. Apply **coordinate update (CU)**

$$x_i^{k+1} = \begin{cases} \left(T(x^k)\right)_i & \text{if } i = i_k \\ x_i^k & \text{otherwise.} \end{cases}$$

Overview

1. Re-formulate a problem into

$$x = T(x)$$

   (use operator splitting)

2. Apply **coordinate update (CU)**

$$x_i^{k+1} = \begin{cases} \left(T(x^k)\right)_i & \text{if } i = i_k \\ x_i^k & \text{otherwise.} \end{cases}$$

3. **Parallelize CU** without sync or locking

Primal-dual splitting by example

- **convex problem**:

$$\underset{x \in \mathcal{H}}{\text{minimize}} \;\; f(x) + g(x) + h(Ax)$$

  only f is differential, g, h are possible nondifferentiable

- optimality condition: $0 \in \nabla f(x) + \partial g(x) + A^T \partial h(Ax)$

Primal-dual splitting by example

- **convex problem**:

$$\underset{x \in \mathcal{H}}{\text{minimize}} \;\; f(x) + g(x) + h(Ax)$$

  only f is differential, g, h are possible nondifferentiable

- optimality condition: $0 \in \nabla f(x) + \partial g(x) + A^T \partial h(Ax)$

- introduce $s \in \partial h(Ax)$; by Fenchel-Young, $Ax \in \partial h^*(s)$

Primal-dual splitting by example

- **convex problem**:

$$\underset{x\in\mathcal{H}}{\text{minimize}} \;\; f(x) + g(x) + h(Ax)$$

  only f is differential, g, h are possible nondifferentiable

- optimality condition: $0 \in \nabla f(x) + \partial g(x) + A^T \partial h(Ax)$

- introduce $s \in \partial h(Ax)$; by Fenchel-Young, $Ax \in \partial h^*(s)$

- **equivalent primal-dual inclusion**:

$$0 \in \Bigg(\underbrace{\begin{bmatrix} \nabla f & 0 \\ 0 & 0 \end{bmatrix}}_{\text{operator } \mathcal{A}} + \underbrace{\begin{bmatrix} \partial g & \\ & \partial h^* \end{bmatrix} + \begin{bmatrix} 0 & A^\top \\ -A & 0 \end{bmatrix}}_{\text{operator } \mathcal{B}} \Bigg) \underbrace{\begin{bmatrix} x \\ s \end{bmatrix}}_{z},$$

Operator splitting

- **forward-backward splitting**: let $U = U^T \succ 0$, $\gamma > 0$

$$\begin{aligned} 0 \in (\mathcal{A} + \mathcal{B})z &\Leftrightarrow (U - \gamma\mathcal{A})z \in (U + \gamma\mathcal{B})z \\ &\Leftrightarrow (U + \gamma\mathcal{B})^{-1}(U - \gamma\mathcal{A})z = z \end{aligned}$$

Operator splitting

- **forward-backward splitting**: let $U = U^T \succ 0$, $\gamma > 0$

$$\begin{aligned} 0 \in (\mathcal{A} + \mathcal{B})z &\Leftrightarrow (U - \gamma\mathcal{A})z \in (U + \gamma\mathcal{B})z \\ &\Leftrightarrow (U + \gamma\mathcal{B})^{-1}(U - \gamma\mathcal{A})z = z \end{aligned}$$

- **abstract fixed-point iteration**:

$$z^{k+1} = \underbrace{(U + \gamma\mathcal{B})^{-1}(U - \gamma\mathcal{A})}_{T} z^k$$

converges for proper γ; diverge unboundedly if T has no fixed-point

Operator splitting

- **forward-backward splitting**: let $U = U^T \succ 0$, $\gamma > 0$

$$\begin{aligned} 0 \in (\mathcal{A} + \mathcal{B})z &\Leftrightarrow (U - \gamma\mathcal{A})z \in (U + \gamma\mathcal{B})z \\ &\Leftrightarrow (U + \gamma\mathcal{B})^{-1}(U - \gamma\mathcal{A})z = z \end{aligned}$$

- **abstract fixed-point iteration**:

$$z^{k+1} = \underbrace{(U + \gamma\mathcal{B})^{-1}(U - \gamma\mathcal{A})}_{T} z^k$$

converges for proper γ; diverge unboundedly if T has no fixed-point

- **algorithm**: set a proper U, Condat-Vu iteration (Chambolle-Pock in imaging)

$$\begin{cases} s^{k+1} = \mathbf{prox}_{\gamma h^*}(s^k + \gamma A x^k), \\ x^{k+1} = \mathbf{prox}_{\eta g}(x^k - \eta(\nabla f(x^k) + A^T(2s^{k+1} - s^k))), \end{cases}$$

Primal-dual splitting for extended monotropic program

$$\begin{aligned} \underset{x_1,\ldots,x_m}{\text{minimize}} \;\; & g_1(x_1) + g_2(x_2) + \cdots + g_m(x_m) + f(x) \\ \text{subject to} \;\; & A_1x_1 \; + \; A_2x_2 \; + \cdots + \; A_mx_m \; = b \end{aligned}$$

- function f is smooth and nonseparable
- **primal-dual splitting with a special metric:** a new parallel algorithm

$$\begin{cases} s^{k+1} = s^k + \gamma(Ax^k - b) \\ x_i^{k+1} = \mathbf{prox}_{\eta g_i}\Big(x_i^k - \eta(\nabla_i f(x^k) + A_i^T s^k + 2\gamma A_i^T Ax^k - 2\gamma A_i^T b)\Big), \;\; i = 1, \ldots, m \end{cases}$$

  - **Jacobi style**
  - Gives async-parallel algorithms for: LP, QP, SOCP, multi-block ADMM-able problems (no matrix inversions)...

Coordinate update

Original (full) iteration: $z \in \mathbb{R}^m$,

$$z^{k+1} = Tz^k$$

Original (full) iteration: $z \in \mathbb{R}^m$,

$$z^{k+1} = Tz^k$$

Coordinate update (CU) iteration:

- iteration: pick a coordinate $i_k \in [m]$

$$z_i^{k+1} = \begin{cases} (Tz^k)_i, & \text{if } i = i_k \\ z_i^k, & \text{otherwise.} \end{cases}$$

(may require stepsize reduction)

Coordinate friendly structure

- CU is fast only if the subproblems are simple

[1] Z. Peng, T. Wu, Y. Xu, M. Yan, and W. Yin, *Coordinate Friendly Structures, Algorithms and Applications*, Annals of Math Sci. and App., 2016.

Coordinate friendly structure

- CU is fast only if the subproblems are simple
- **require**: $\mathrm{cost}\big[\text{each } \mathrm{CU}_i(z)\big] = O\big(\frac{1}{m}\mathrm{cost}[z \mapsto Tz]\big)$ for all z, i

[1] Z. Peng, T. Wu, Y. Xu, M. Yan, and W. Yin, *Coordinate Friendly Structures, Algorithms and Applications*, Annals of Math Sci. and App., 2016.

Coordinate friendly structure

- CU is fast only if the subproblems are simple
- **require**: $\mathrm{cost}\big[\text{each } \mathrm{CU}_i(z)\big] = O\big(\frac{1}{m}\mathrm{cost}[z \mapsto Tz]\big)$ for all z, i
- Found on most first-order (splitting) methods for structured problems[1]

[1] Z. Peng, T. Wu, Y. Xu, M. Yan, and W. Yin, *Coordinate Friendly Structures, Algorithms and Applications*, Annals of Math Sci. and App., 2016.

Coordinate friendly structure

- CU is fast only if the subproblems are simple
- **require**: $\mathrm{cost}\big[\text{each } \mathrm{CU}_i(z)\big] = O\big(\frac{1}{m}\mathrm{cost}[z \mapsto Tz]\big)$ for all z, i
- Found on most first-order (splitting) methods for structured problems[1]
- May require caching and maintaining extra variables
- May require switch operator orders in splitting

[1] Z. Peng, T. Wu, Y. Xu, M. Yan, and W. Yin, *Coordinate Friendly Structures, Algorithms and Applications*, Annals of Math Sci. and App., 2016.

Convergence overview

- Aim for: **minimal assumptions** and **generality**
- Typically without: strongly monotone, strongly convex, bounded set

[2] Combettes, Pesquet'15

Convergence overview

- Aim for: **minimal assumptions** and **generality**
- Typically without: strongly monotone, strongly convex, bounded set
- KM'50s iteration $z^{k+1} = Tz^k$ converges if
  - T has a fixed-point
  - $T = (1-\alpha)I + \alpha T'$ for some nonexpansive T' and $\alpha \in (0, 1)$

[2] Combettes, Pesquet'15

Convergence overview

- Aim for: **minimal assumptions** and **generality**
- Typically without: strongly monotone, strongly convex, bounded set
- KM'50s iteration $z^{k+1} = Tz^k$ converges if
  - T has a fixed-point
  - $T = (1-\alpha)I + \alpha T'$ for some nonexpansive T' and $\alpha \in (0, 1)$
- Parallel-CU (synchronous): identical to KM

[2] Combettes, Pesquet'15

Convergence overview

- Aim for: **minimal assumptions** and **generality**
- Typically without: strongly monotone, strongly convex, bounded set
- KM'50s iteration $z^{k+1} = Tz^k$ converges if
  - T has a fixed-point
  - $T = (1-\alpha)I + \alpha T'$ for some nonexpansive T' and $\alpha \in (0, 1)$
- Parallel-CU (synchronous): identical to KM
- Random-CU[2]: take conditional expectation and apply *super-martingale convergence*

[2] Combettes, Pesquet'15

Convergence overview

Convergence under *diagonal dominance*:

- Linear Gauss-Seidel converges if $\rho(A) < 1$

Convergence overview

Convergence under *diagonal dominance*:

- Linear Gauss-Seidel converges if $\rho(A) < 1$
- Baudet'78 introduces *absolute-contraction* conditions for general T
- *Absolute-contraction* is sufficient for asynchronous CU (with different details for partial and total versions)

Convergence overview

Convergence under *diagonal dominance*:

- Linear Gauss-Seidel converges if $\rho(A) < 1$
- Baudet'78 introduces *absolute-contraction* conditions for general T
- *Absolute-contraction* is sufficient for asynchronous CU (with different details for partial and total versions)
- Comprehensive treatment: Bertsekas-Tsitsiklis'89 and Frommer-Szyld'00

Convergence overview

Convergence under *diagonal dominance*:

- Linear Gauss-Seidel converges if $\rho(A) < 1$
- Baudet'78 introduces *absolute-contraction* conditions for general T
- *Absolute-contraction* is sufficient for asynchronous CU (with different details for partial and total versions)
- Comprehensive treatment: Bertsekas-Tsitsiklis'89 and Frommer-Szyld'00
- Some but not many problems satisfy the assumptions

Our results

Convergence under *nonexpansiveness*

- Cyclic CU converges under sufficiently step size $O(\frac{1}{\sqrt{k}})$ or $O(\frac{1}{mM})$

Our results

Convergence under *nonexpansiveness*

- Cyclic CU converges under sufficiently step size $O(\frac{1}{\sqrt{k}})$ or $O(\frac{1}{mM})$
- Async-parallel CU converges under
  - sufficiently step size $O(\frac{1}{1+\tau/\sqrt{m}})$
  - random coordinate selection
  - delay is independent of coordinate selection
- Allow unbounded delays with either $o(1/k^3)$ tail or finite $\liminf$

Our results

Convergence under *nonexpansiveness*

- Cyclic CU converges under sufficiently step size $O(\frac{1}{\sqrt{k}})$ or $O(\frac{1}{mM})$
- Async-parallel CU converges under
  - sufficiently step size $O(\frac{1}{1+\tau/\sqrt{m}})$
  - random coordinate selection
  - delay is independent of coordinate selection
- Allow unbounded delays with either $o(1/k^3)$ tail or finite $\liminf$
- In practice, (shuffled) cyclic CU performs better

Our results

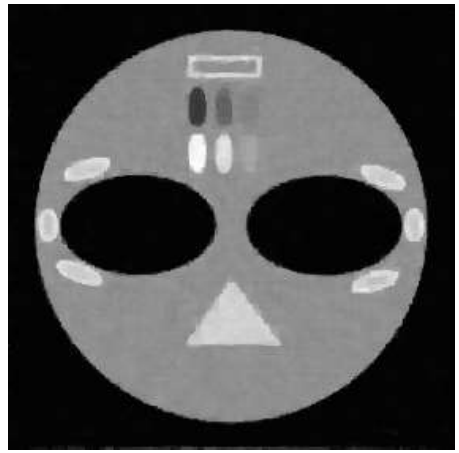
Convergence under *nonexpansiveness*

- Cyclic CU converges under sufficiently step size $O(\frac{1}{\sqrt{k}})$ or $O(\frac{1}{mM})$
- Async-parallel CU converges under
  - sufficiently step size $O(\frac{1}{1+\tau/\sqrt{m}})$
  - random coordinate selection
  - delay is independent of coordinate selection
- Allow unbounded delays with either $o(1/k^3)$ tail or finite $\liminf$
- In practice, (shuffled) cyclic CU performs better
- Small coordinates require non-Matlab (i.e. C) implementations

CT recovery: full *vs* random coordinate update

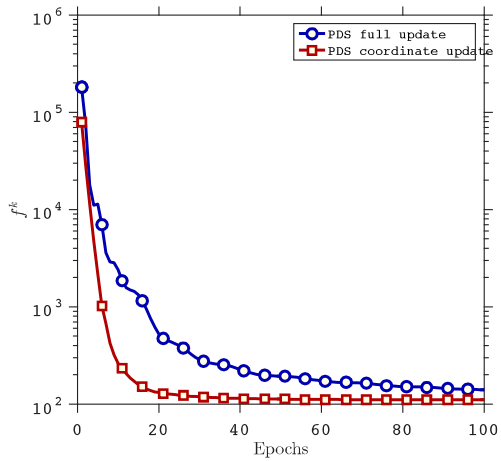
- 284×284, 90 beam projections, 362 measurements each beam
- TV minimization partitioned to 284 columns (blocks), run 100 epochs

(a) Phantom image



(b) Recovered by PDS

(c) Recovered by PDS coord



(d) Objective function value

Portfolio optimization simulation

$$\begin{array}{ll}\text{minimize}_x & \underbrace{\frac{1}{2}x^\top Q x}_{\text{risk}}, \\ \text{subject to} & \underbrace{x \geq 0}_{\text{buy}}, \quad \underbrace{\sum_{i=1}^m x_i \leq 1}_{\text{limit}}, \quad \underbrace{\sum_{i=1}^m \xi_i x_i \geq c}_{\text{return}},\end{array}$$

- **reformulation**:

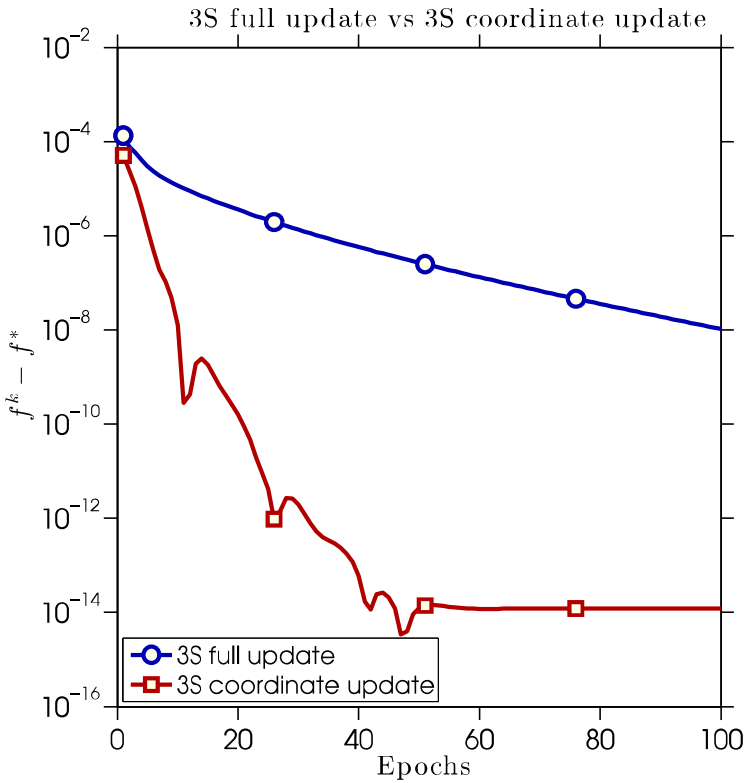
$$\underset{x}{\text{minimize}} \;\; \text{risk} + \iota_{D_1}(x) + \iota_{D_2}(x)$$

  where $D_1 = \{\text{buy}\}$ and $D_2 = \{\text{limit}, \text{return}\}$

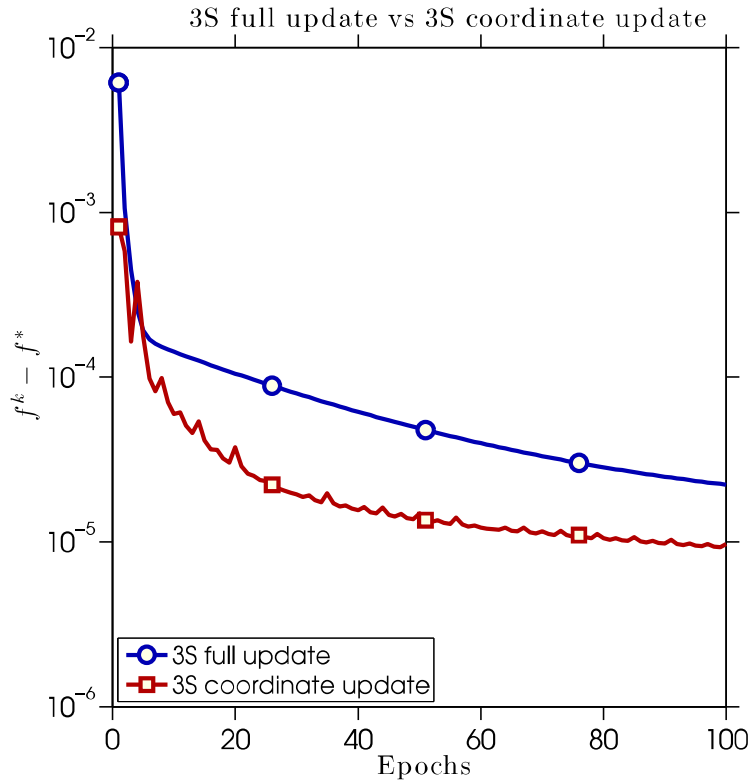
- full update based on 3-splitting operator (Davis-Y.'15)

$$z^{k+1} = T_{3S}(z^k)$$

Full *vs* random coordinate update



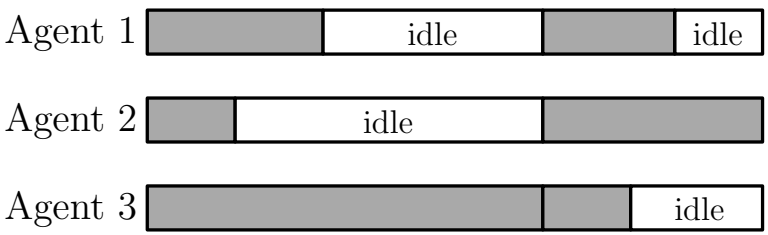
(a) Synthesis dataset



(b) NASDAQ dataset

Parallelization

Sync-parallel versus async-parallel



Synchronous

(wait for the slowest)

Asynchronous

(non-stop, no wait)

ARock[3]: Async-parallel coordinate update

- $x = (x_1, \ldots, x_m) \in \mathcal{H}_1 \times \cdots \times \mathcal{H}_m$
- let $S_i(x) := x_i - (T(x))_i$
- **algorithm:** each agent randomly picks $i_k \in \{1, \ldots, m\}$:

$$x_i^{k+1} \leftarrow \begin{cases} x_i^k - \eta_k S_i(x^{k-d_k}), & \text{if } i = i_k \\ x_i^k, & \text{otherwise.} \end{cases}$$

- **assumptions:** atomic update, delay independent of i_k
- **guarantee:** provably weakly converges a.s. under proper η_k

[3] Peng-Xu-Yan-Y. SISC'16

TMAC: A Toolbox of Async-Parallel, Coordinate, Splitting, and Stochastic Methods

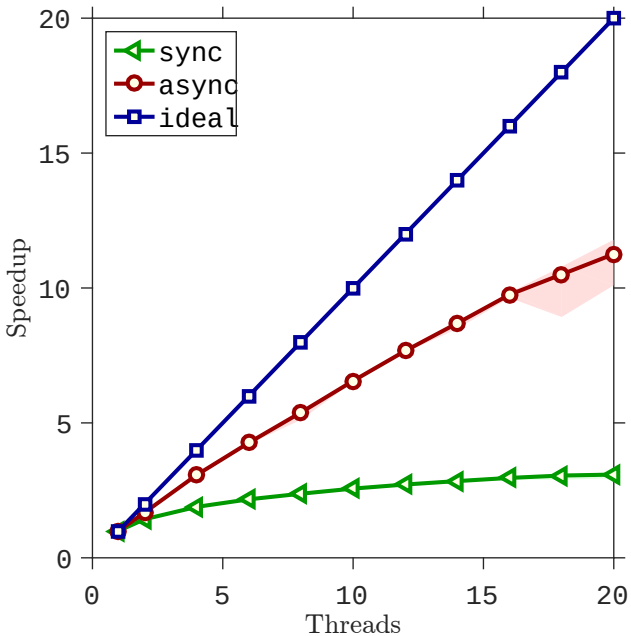
- C++11 multi-threading (no shared-memory parallelism in Matlab)
- Plug in your operators, get free coordinate-update and async-parallelism
- `github.com/uclaopt/tmac`
  - committers: Brent Edmunds, Zhimin Peng
  - contributors: Yerong Li, Yezheng Li, Tianyu Wu
  - supports: Windows, Mac, Linux

ℓ_1 logistic regression

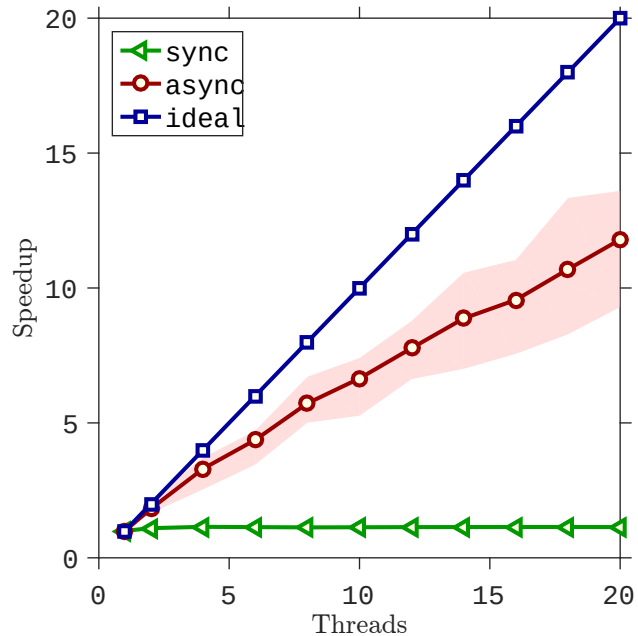
- **model**:

$$\underset{x\in\mathbb{R}^n}{\text{minimize}}\ \lambda\|x\|_1 + \frac{1}{N}\sum_{i=1}^{N}\log\left(1+\exp(-b_i\cdot a_i^T x)\right), \tag{1}$$

- sparse numerical linear algebra are used for datasets: news20, url



dataset "news20"



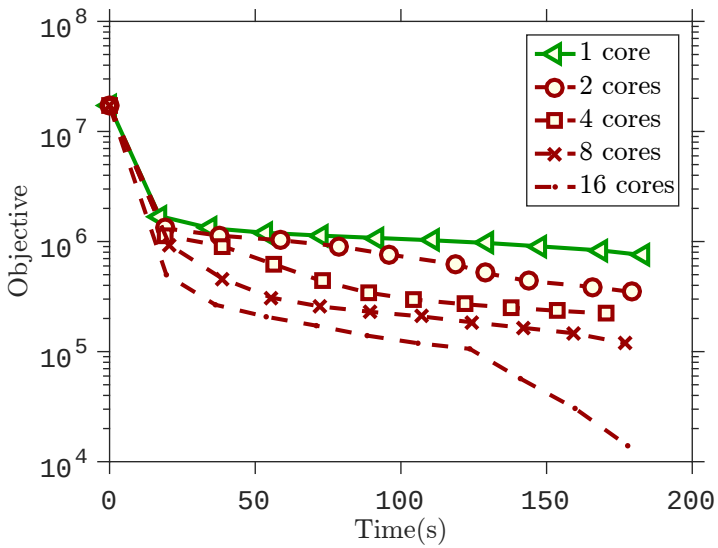
dataset "url"

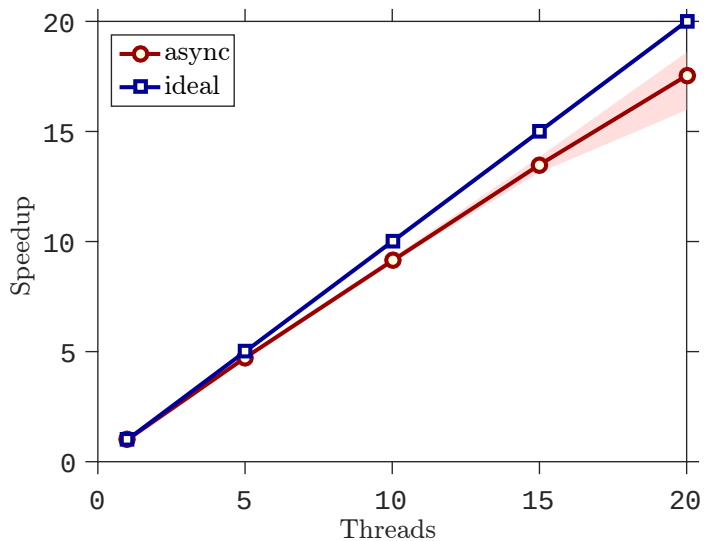
Nonnegative matrix factorization

- **model**:

$$\underset{X,Y \geq 0}{\text{minimize}} \ \|A - X^T Y\|_F^2, \tag{2}$$

- despite nonconvex, amenable to parallel coordinate descent





Summary of coordinate update:

- applicable to many problems
- faster than full update
- scalable, (asynchronously) parallelizable

Papers: UCLA CAM 15-37 (theory), 16-13 (applications), 16-37 (solver)

Code: `github.com/uclaopt/tmac`

Sparse Recovery by Differential Inclusion

Wotao Yin

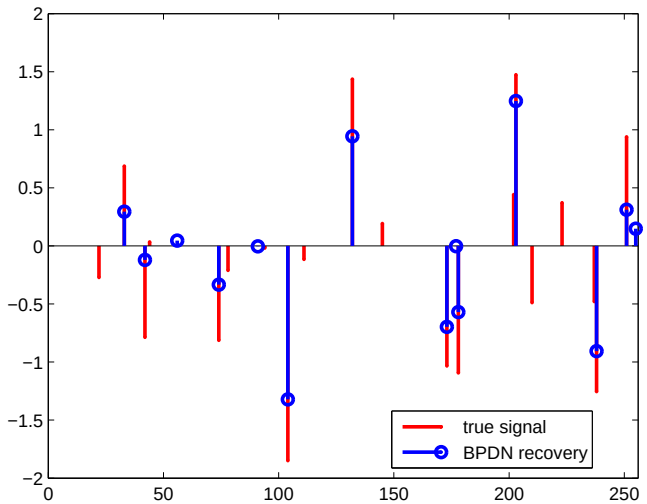
Joint with: Stanley Osher, Ming Yan (UCLA)

Feng Ruan, Jiechao Xiong, Yuan Yao (Peking U)

SAMSI OPT Opening Workshop — August 2016

A synthetic example

- Setup: $m = 25$, $n = 256$, gaussian noise ϵ



True vs LASSO (hand tuned)

Express the bias

- LASSO:

$$x^{\text{lasso}} \leftarrow \underset{x}{\text{minimize}} \, \|x\|_1 + \frac{t}{2m}\|Ax - b\|_2^2$$

- Solution satisfies:

$$0 = p + \frac{t}{m} A^T (Ax^{\text{lasso}} - b) \quad \text{where } p \in \partial \|x^{\text{lasso}}\|_1$$

- Assume obtaining the true support $S = \text{supp}(x^*) = \text{supp}(x^{\text{lasso}})$

$$x_S^{\text{lasso}} = \underbrace{x_S^* + \frac{1}{m}(A_S^T A_S)^{-1} A_S^T \epsilon}_{\textbf{oracle estimate, } \mathbf{E}(\ldots) = x_S^*} \underbrace{-\frac{1}{t}(A_S^T A_S)^{-1} \text{sign}(x_S^{\text{lasso}})}_{\textbf{bias}}.$$

Idea: remove LASSO bias at its origin

- Recall LASSO satisfies: exists $p \in \partial \|x^{\text{lasso}}\|_1$ such that

$$\boldsymbol{p} = \frac{\boldsymbol{t}}{m} A^T (b - A\boldsymbol{x}^{\text{lasso}})$$

- $\frac{d}{dt}$ yields

$$\dot{p} = \frac{1}{m} A^T (b - A(t\dot{x}^{\text{lasso}} + x^{\text{lasso}}))$$

- Observe: $t\dot{x}^{\text{lasso}} + x^{\text{lasso}}$ is the LASSO solution with the bias removed

Idea: remove LASSO bias at its origin

- Recall LASSO satisfies: exists $p \in \partial \|x^{\text{lasso}}\|_1$ such that

$$\boldsymbol{p} = \frac{\boldsymbol{t}}{m} A^T (b - A\boldsymbol{x}^{\text{lasso}})$$

- $\frac{d}{dt}$ yields

$$\dot{p} = \frac{1}{m} A^T (b - A(t\dot{x}^{\text{lasso}} + x^{\text{lasso}}))$$

- Observe: $t\dot{x}^{\text{lasso}} + x^{\text{lasso}}$ is the LASSO solution with the bias removed
- Replacing $t\dot{x}^{\text{lasso}} + x^{\text{lasso}}$ by x yields the ISS dynamic

$$\dot{p} = \frac{1}{m} A^T (b - Ax)$$

Sparse recovery by differential inclusion

- Name from image processing
- Solution path $\{x(t), p(t)\}_{t \geq 0}$ to

$$\dot{p}(t) = \underbrace{\frac{1}{m} A^T (b - Ax(t))}_{-\nabla \text{fitting}},$$

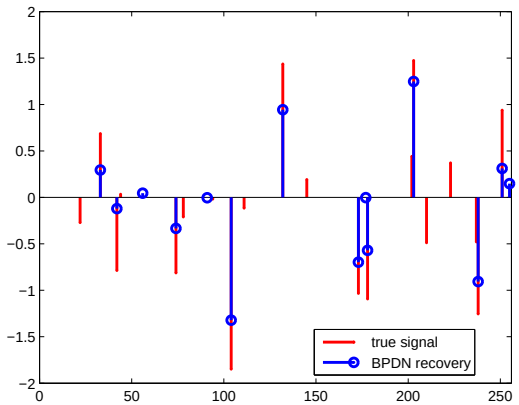
$$p(t) \in \partial \|x(t)\|_1.$$

from initial $x(0) = p(0) = 0$

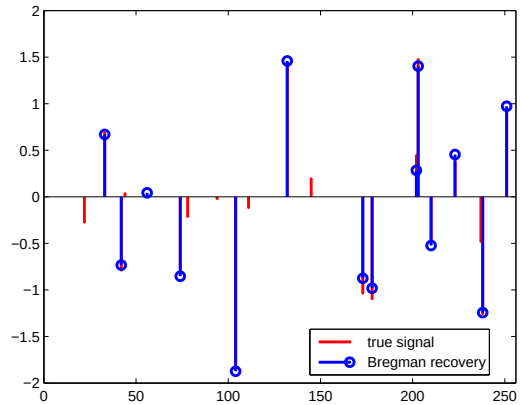
  - $p(t)$ is *unique, piece-wise linear*, and
  - $x(t)$ is *piece-wise constant*.

Compare to LASSO

Apply them to the same synthetic example



True vs LASSO
(shown previously)



True vs ISS

Numerical result: prostate tumor size

- the first example from Hastie-Tibshirani-Friedman
- problem: given 8 clinical features, select predictors for prostate tumor size
  - data: 67 training cases + 30 testing cases;
    - parameters picked by cross validation

Predictor	**LS**	**Subset Selection**	**LASSO**	**ISS**
Intercept	2.452	2.466	2.481	2.476
lcavol	0.716	0.667	0.622	0.554
lweight	0.293	0.366	0.289	0.279
age	-0.143	0	-0.096	0
lbph	0.212	0	0.188	0.198
svi	0.310	0.268	0.262	0.238
lcp	-0.289	-0.291	-0.164	0
gleason	-0.021	0	0	0
pgg45	0.277	0.227	0.187	0.122
#predictors	8	**5**	7	**5**
Test error	0.586	0.587	0.543	**0.541**

Summary

- ISS improves over LASSO and slightly over debiased-LASSO
- **Paper:** Sparse Recovery by Differential Inclusion. ACHA'16
- **Computing:** exact algorithm, inexact algorithm (linearized Bregman)
  google "linearized bregman matlab"
  "Libra" at cran by F. Ruan, J. Xiong and Y. Yao
- Related to Genevera Allen's talk on Friday