Asynchronous parallel primal-dual block update method

Yangyang Xu

Department of Mathematical Sciences, RPI

MOPTA 2017, Lehigh University

August 18, 2017

Multi-block affinely constrained programs

$$\min_x F(x) = f(x_1, \ldots, x_m) + \sum_{i=1}^{m} g_i(x_i), \text{ s.t. } \sum_{i=1}^{m} A_i x_i = b. \qquad \text{(MBP)}$$

- $x = (x_1, \ldots, x_m)$: partition depends on applications
- f Lipschitz differentiable; g_i simple and possibly nondifferentiable
- Examples: basis pursuit, dual support vector machine, portfolio optimization, ...

This talk: one parallel method for (MBP) **based on block update**

Proximal block coordinate update method

Without affine constraint: $\min_x f(x) + \sum_{i=1}^m g_i(x_i)$.

Algorithm 1: Proximal block coordinate update with initial point x^0

for $k = 0, 1, \ldots,$ **do**

Choose one block i_k.

For $i \neq i_k$, let $x_i^{k+1} = x_i^k$ and for $i = i_k$, update x_i by

$$x_i^{k+1} \in \arg\min_{x_i} \langle \nabla_{x_i} f(x^k), x_i \rangle + \frac{\eta_i}{2} \|x_i - x_i^k\|^2 + g_i(x_i).$$

- Different ways to choose i_k: cyclic, random, greedy
- First appears in [Tseng-Yun'09] and popularized since [Nesterov'12]
- Convergence and rate extensively studied [Nesterov'12, Richtarik-Takac'14, Razaviyayn-Hong-Luo'13, X.-Yin'13, X.-Yin'17, Hong et. al'17, etc]
- **May not converge for nondifferentiable nonseparable function**

Alternating direction method of multipliers

Without nonseparable term f: $\min_x \sum_{i=1}^m g_i(x_i), \text{ s.t. } \sum_{i=1}^m A_i x_i = b$

Algorithm 2: Alternating direction method of multipliers with initial x^0, λ^0

for $k = 0, 1, \ldots,$ **do**

Update x_i for $i = 1, 2, \ldots, m$ sequentially by

$$x_i^{k+1} \in \arg\min_{x_i} g_i(x_i) + \langle \lambda^k, A_i x_i \rangle + \frac{\beta}{2} \left\| A_{<i} x_{<i}^{k+1} + A_i x_i + A_{>i} x_{>i}^k - b \right\|^2.$$

Let $\lambda^{k+1} \leftarrow \lambda^k + \rho(Ax^{k+1} - b)$.

- Simpler than the augmented Lagrangian method
- 2-block convex programs: convergence guaranteed, also known as Douglas-Rachford splitting [DR'56]
- 3 or more block convex programs: may diverge [Chen et. al'16]

Divergence of 3-block ADMM [Chen et. al'16]

Apply ADMM to $\{\min 0, \ \text{s.t.}\ A_1 x_1 + A_2 x_2 + A_3 x_3 = 0\}$

- Equivalent to the iterative method: $\begin{bmatrix} x_2^{k+1} \\ x_3^{k+1} \\ \boldsymbol{\lambda}^{k+1} \end{bmatrix} = \mathbf{L}^{-1}\mathbf{R} \begin{bmatrix} x_2^{k} \\ x_3^{k} \\ \boldsymbol{\lambda}^{k} \end{bmatrix}$, where

$$\mathbf{L} = \begin{bmatrix} A_2^\top A_2 & 0 & \mathbf{0} \\ A_3^\top A_2 & A_3^\top A_3 & \mathbf{0} \\ A_2 & A_3 & \mathbf{I} \end{bmatrix}$$

and

$$\mathbf{R} = \begin{bmatrix} 0 & -A_2^\top A_3 & A_2^\top \\ 0 & 0 & A_3^\top \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} - \frac{1}{\|A_1\|^2} \begin{bmatrix} A_2^\top A_1 \\ A_3^\top A_1 \\ A_1 \end{bmatrix} \left[-A_1^\top A_2, -A_1^\top A_3, A_1^\top\right]$$

Divergence of 3-block ADMM [Chen et. al'16]

- Guaranteed convergence if $\rho(\mathbf{L}^{-1}\mathbf{R}) < 1$ and divergence if $\rho(\mathbf{L}^{-1}\mathbf{R}) > 1$
- Let $\mathbf{A} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 2 & 2 \end{bmatrix}$
- $\rho(\mathbf{L}^{-1}\mathbf{R}) \approx 1.0278$ and thus diverges

Remidies for convergence of multi-block ADMM

Additional assumptions

- parts of the objective are strongly convex [Han-Yuan'12, Chen-Shen-You'13, Cai-Han-Yuan'17]
- near orthogonality between A_i's [Chen et. al'16]

Modification

- back substitution after each cycle [He-Tao-Yuan'12]
- Jacobi proximal update [He-Ma-Yuan'16, Deng et. al'17]
- random permutation before each cycle and on linear system [Sun-Luo-Ye'15]

Proposed: primal-dual randomized block update

Algorithm 3: Primal-dual randomized block update method

Initialization: choose x^0, λ^0 and let $r^0 = Ax^0 - b$.

for $k = 0, 1, \ldots,$ **do**

Choose i_k uniformly at random.

For $i \neq i_k$, let $x_i^{k+1} = x_i^k$ and for $i = i_k$, update x_i by

$$x_i^{k+1} \in \arg\min_{x_i} \left\langle \nabla_{x_i} f(x^k) + A_i^\top (\lambda^k + \beta r^k), x_i \right\rangle + \frac{\eta_i}{2} \|x_i - x_i^k\|^2 + g_i(x_i)$$

Let $r^{k+1} \leftarrow r^k + A_{i_k}(x_{i_k}^{k+1} - x_{i_k}^k)$ and $\lambda^{k+1} \leftarrow \lambda^k + \rho(Ax^{k+1} - b)$.

- Similar idea appears in [Dang-Lan'14, Yu-Lin-Yang'15] for bilinear saddle-point problems without nonseparable f
- [Hong et. al'14] randomly picks one among x_i's and λ; subsequence convergence if dual stepsize diminishing

Gao-X.-Zhang'16. Randomized Primal-Dual Proximal Block Coordinate Updates.

Convergence results

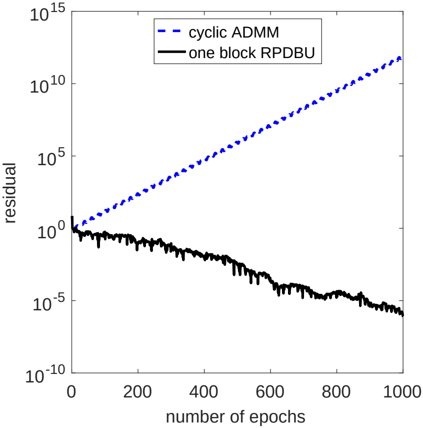
Assumptions:

1. Existence of a primal-dual solution (x^*, λ^*)
2. f and g_i's are convex
3. $\nabla_{x_i} f$ Lipschitz continuous with constant L_i

Convergence results: if $\eta_i \geq L_i + \beta\|A_i\|^2$ and $\rho \in (0, \frac{\beta}{m}]$, then

- $F(x^k) \xrightarrow{p} F(x^*)$ and $\|r^k\| \xrightarrow{p} 0$
- $\max\left(|\mathbb{E}[F(\bar{x}^k) - F(x^*)]|, \mathbb{E}\|A\bar{x}^k - b\|\right) = O(1/k)$ with $\bar{x}^k$ averaged point

Test on the counterexample in [Chen et. al'16]

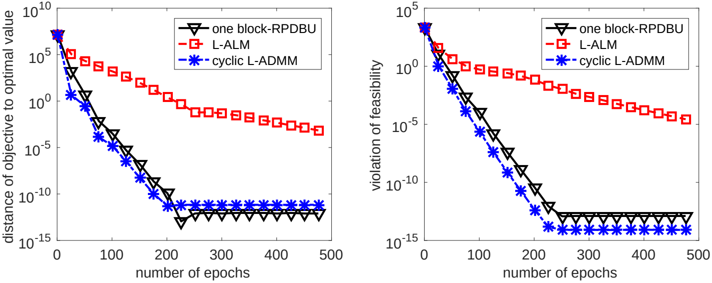


Test on quadratic programming

Nonnegativity constrained quadratic programming:

$$\underset{x}{\text{minimize}}\ \frac{1}{2}x^\top Qx + c^\top x, \quad \text{subject to } Ax = b,\ x \geq 0.$$

- Q positive semidefinite and singular



Coordinate friendly structure

An operator T is coordinate friendly if

$$\sum_{i=1}^{m} \text{cost}[(Tx)_i] = O\left(\text{cost}[Tx]\right)$$

- May require maintaining intermediate computation

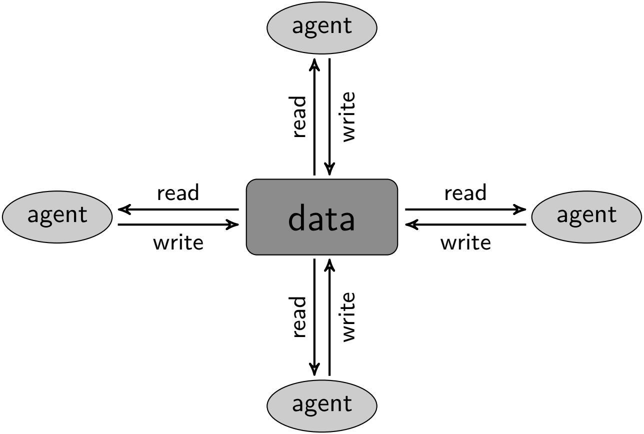
Examples:

- Matrix-vector multiplication: $T(x) = Ax + b$
- Proximal mapping of separable function $f(x) = \sum_{i=1}^{m} f_i(x_i)$
- Gradient of regression functions e.g., least square, logistic regression
- More ...

Peng-Wu-X.-Yan-Yin'16. Coordinate update methods: structures, algorithms, and applications. *Annals of Mathematical Sciences and Applications*.

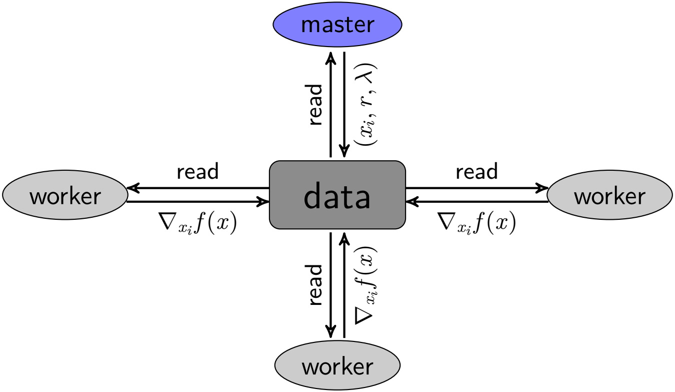
Async-parallel primal-dual block update

Architecture



- data stored in a global memory
- each agent (CPU, core) can read and write data from the global memory

Diagram of proposed async-parallel method



- Worker nodes compute partial gradients $\nabla_{x_i} f(x)$
- Master node updates x and λ

Pseudocode

Algorithm 4: Async-parallel primal-dual block update method

Initialization: choose x^0 and λ^0; let $r^0 = Ax^0 - b$ and $k = 0$.

while *Not convergent* **do**

- **if** *worker node* **then**
  - Pick j uniformly at random and read x as $\hat{x}$.
  - Compute $\nabla_j f(\hat{x})$ and write it together with j to global memory
- **if** *master node* **then**
  - **if** *one new pair* $\left(j, \nabla_j f(\hat{x})\right)$ **then**
    - Let $i_k = j$ and $v^k = \nabla_j f(\hat{x})$
  - **else**
    - Pick i_k uniformly at random and let $v^k = \nabla_{i_k} f(x^k)$
  - For any $i \neq i_k$, keep $x_i^{k+1} = x_i^k$, and for $i = i_k$, update x_i by

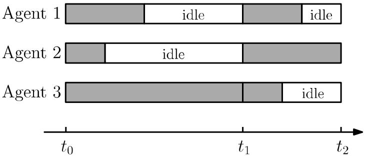
$$x_i^{k+1} \in \arg\min_{x_i} \langle v^k - A_i^\top(\lambda^k - \beta r^k), x_i \rangle + g_i(x_i) + \frac{\eta_i}{2}\|x_i - x_i^k\|^2.$$

  - Update r and λ; increase k.

- all nodes continuously working
- Delay on x: used $\hat{x}$ may not be the current iterate in the memory
- No delay on r and λ
- No waste of partial gradients
  - if $\nabla_{x_i} f$ much more expensive than $A_i^\top \lambda$ and $\mathbf{prox}_{g_i}$
  - examples: dual support vector machine, portfolio optimization

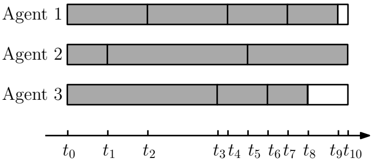
Sync-parallel versus Async-parallel

Synchronous



- Pros: convergence directly from non-parallel method
- Cons: idle time wasted, memory-access congestion

Asynchronous



- Pros: no idle time, no memory-access congestion
- Cons: convergence requires new technique

Convergence results

Assumptions:

1. Existence of a primal-dual solution (x^*, λ^*)
2. f and g_i's are convex
3. $\nabla_{x_i} f$ Lipschitz continuous with constant L_i
4. delay on $\hat{x}$ uniformly bounded by τ

Convergence results: if $\eta_i \geq L_i + \beta\|A_i\|^2 + C\tau^2/m$ and $\rho \in (0, \frac{\beta}{m}]$, then

- $F(x^k) \xrightarrow{p} F(x^*)$ and $\|r^k\| \xrightarrow{p} 0$
- $\max\left(\mathbb{E}|F(\bar{x}^k) - F(x^*)|, \mathbb{E}\|A\bar{x}^k - b\|\right) = O(1/k)$ with $\bar{x}^k$ averaged point

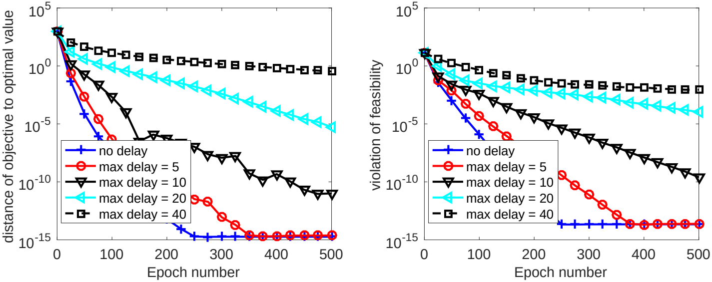
Remark: if $\tau = o(\sqrt{m})$, the stepsize weakly affected by delay and nearly linear speed-up can be achieved

Literature

- Async-parallel method first appears in [Chazan-Miranker'69] for solving linear system
- Recently lots of work on stochastic gradient [Nedic-Bertsekas-Borkar'01, Langford-Smola-Zinkevich'09, Agarwal-Duchi'11, Recht et. al'11, ...]
- Async-parallel randomized block update first in [Liu et. al'15]
- Async-parallel primal-dual block update in [Wei-Ozdaglar'13, Zhang-Kwok'14, Chang et. al'16] for multi-agent optimization
- Async-parallel method for fixed point problems [Baudet'78, Bertsekas'83, Combettes-Eckstein'16, Peng et. al'16, ...]

Numerical examples

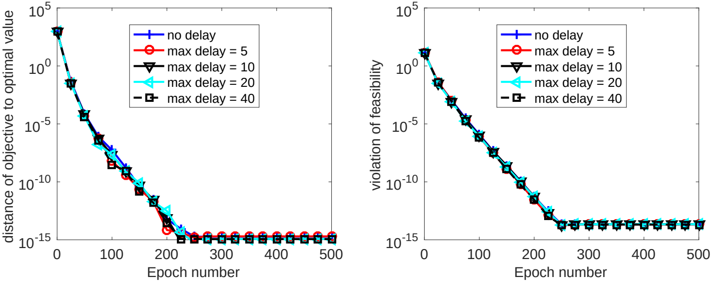
Quadratic programming with delay-dependent stepsize



- simulated results with $m = 2000$
- $\hat{x}$ selected from most recent τ iterates uniformly at random
- stepsize set dependent on delay

Observation: delay affects convergence

Quadratic programming with delay-independent stepsize



- simulated results with $m = 2000$
- $\hat{x}$ selected from most recent τ iterates uniformly at random
- stepsize set regardless of delay

Observation: delay almost not affects convergence

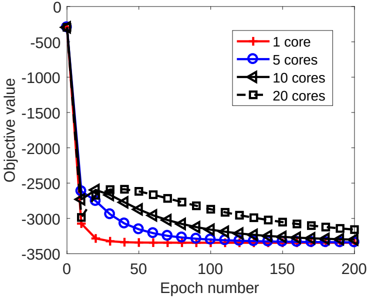
Dual support vector machine

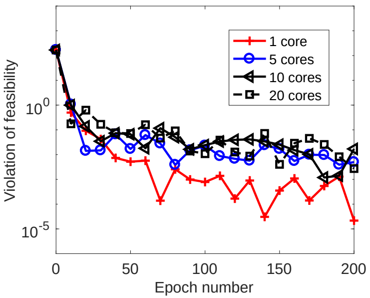
$$\boxed{\min_{\theta} \frac{1}{2}\theta^\top \mathrm{diag}(y) X^\top X \mathrm{diag}(y)\theta - e^\top \theta, \text{ s.t. } y^\top \theta = 0,\ 0 \le \theta_i \le C, \forall i.}$$

- each column of X one sample
- news20[1] is used and has 19,996 samples and 1,355,191 features
- each block has 50 or 51 coordinates

[1] `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`

Sync-parallel method on dual support vector machine

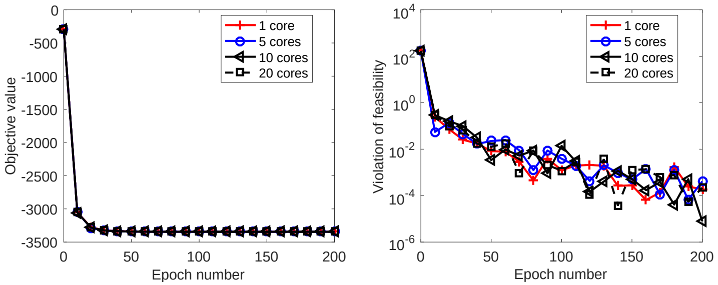




- Every time, p blocks are selected and updated in parallel, where p is the number of cores

Observation: slower convergence with more blocks updated every time due to smaller stepsize

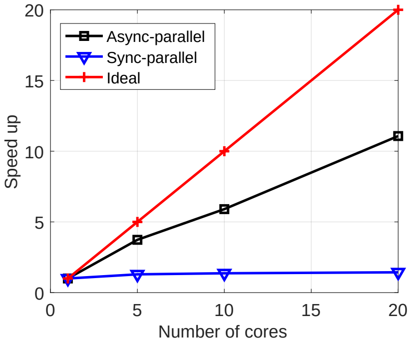
Async-parallel method on dual support vector machine



- stepsize set regardless of delay

Observation: convergence behavior almost same for different numbers of cores

Speed-up by sync and async-parallel methods



- measured by running time

Observation: sync-parallel method almost no speed-up due to inbalance loading and idle waiting time while async-parallel method about x12 speed-up

Conclusions

- Proposed a primal-dual randomized block update method for affinely constrained multi-block convex programs
- Proposed an async-parallel primal-dual block update method
- Established probability convergence and sublinear rate results
- Presented numerical results on both sync and asyn-parallel methods and significantly better speed-up observed for the latter

Open question: convergence and speed up if r and λ outdated

References

1. Y. Xu. *Asynchronous parallel primal-dual block update methods*, arXiv preprint arXiv:1705.06391, 2017.
2. X. Gao, Y. Xu, and S. Zhang. *Randomized Primal-Dual Proximal Block Coordinate Updates*, arXiv preprint arXiv:1605.05969, 2016.
3. Z. Peng, T. Wu, Y. Xu, M. Yan, and W. Yin. *Coordinate Friendly Structures, Algorithms and Applications*, Annals of Mathematical Sciences and Applications 1(1), 57–119, 2016.
4. E. Wei and A. Ozdaglar. *On the $O(1/k)$ convergence of asynchronous distributed alternating direction method of multipliers.* IEEE on GlobalSIP, 551–554, 2013.
5. J. Liu, S. J. Wright, C. Ré, V. Bittorf, and S. Sridhar. *An asynchronous parallel stochastic coordinate descent algorithm*. JMLR, 16: 285–322, 2015.