Randomized Primal-Dual Algorithms for Asynchronous Distributed Optimization

Lin Xiao
Microsoft Research

Joint work with
Adams Wei Yu (CMU), Qihang Lin (University of Iowa)
Weizhu Chen (Microsoft)

Workshop on Large-Scale and Distributed Optimization

Lund Center for Control of Complex Engineering Systems
June 14-16, 2017

Motivation

big data optimization problems

- dataset cannot fit into memory or storage of single computer
- require distributed algorithms with inter-machine communication

origins

- machine learning, data mining, ...
- industry: search, online advertising, social media analysis, ...

goals

- asynchronous distributed algorithms deployable in the cloud
- nontrivial communication and/or computation complexity

Outline

- distributed empirical risk minimization
- randomized primal-dual algorithms with parameter servers
- variance reduction techniques
- DSCOVR algorithms
  (Doubly Stochastic Coordinate Optimization with Variance Reduction)
- preliminary experiments

Empirical risk minimization (ERM)

- popular formulation in supervised (linear) learning

$$\underset{w \in \mathbf{R}^d}{\text{minimize}} \quad P(w) \stackrel{\text{def}}{=} \frac{1}{N} \sum_{i=1}^{N} \phi(x_i^T w, y_i) + \lambda g(w)$$

  - i.i.d. samples: $(x_1, y_1), \ldots (x_N, y_N)$ where $x_i \in \mathbf{R}^d$, $y_i \in \mathbf{R}$
  - loss function: $\phi(\cdot, y)$ convex for every y
  - $g(w)$ strongly convex, e.g., $g(w) = (\lambda/2)\|w\|_2^2$
  - regularization parameter $\lambda \sim 1/\sqrt{N}$ or smaller

- **linear regression**: $\phi(x^T w, y) = (y - w^T x)^2$

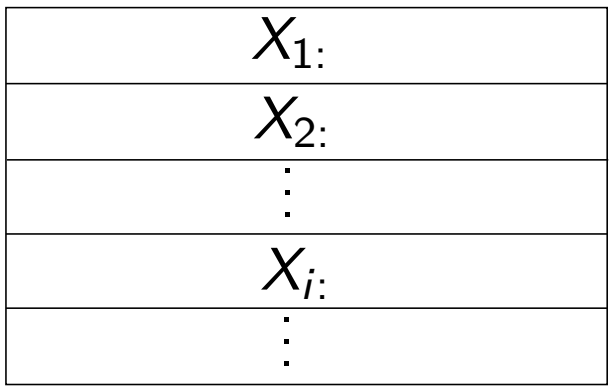
- **binary classification:** $y \in \{\pm 1\}$
  - logistic regression: $\phi(x^T w, y) = \log(1 + \exp(-y(w^T x)))$
  - hinge loss (SVM): $\phi(x^T w, y) = \max\{0,\ 1 - y(w^T x)\}$

Distributed ERM

when dataset cannot fit into memory of single machine

- data partitioned on m machines

$$X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{bmatrix} \in \mathbf{R}^{N \times d}$$



- rewrite objective function

$$\underset{w \in \mathbf{R}^d}{\text{minimize}} \quad \frac{1}{N} \sum_{i=1}^{m} \Phi_i(X_{i:}w) + g(w)$$

where $\Phi_i(X_{i:}w) = \sum_{j \in \mathcal{I}_i} \phi_j(x_j^T w, y_j)$ and $\sum_{i=1}^{m} |\mathcal{I}_i| = N$

Distributed optimization

- **distributed algorithms:** alternate between
  - a local computation procedure at each machine
  - a communication round with simple map-reduce operations (*e.g.*, broadcasting a vector in $\mathbf{R}^d$ to m machines, or computing sum or average of m vectors in $\mathbf{R}^d$)

- **bottleneck:** high cost of inter-machine communication
  - speed/delay, synchronization
  - energy consumption

- **communication-efficiency**
  - number of communication rounds to find $P(\widehat{w}) - P(w^\star) \leq \epsilon$
  - often can be measured by iteration complexity

Iteration complexity

- **assumption:** $f : \mathbf{R}^d - \mathbf{R}$ twice continuously differentiable,

$$\lambda I \preceq f''(w) \preceq LI, \qquad \forall\, w \in \mathbf{R}^d$$

  in other words, f is λ-strongly convex and L-smooth

- **condition number**

$$\kappa = \frac{L}{\lambda}$$

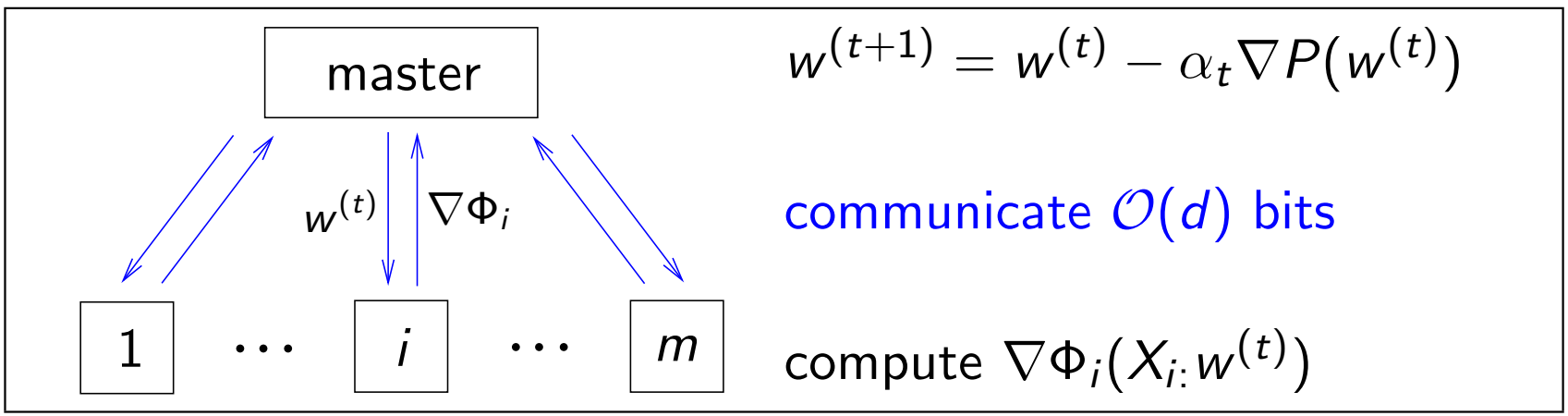
  we focus on ill-conditioned problems: $\kappa \gg 1$

- **iteration complexities** of first-order methods
  - gradient descent method: $\mathcal{O}(\kappa \log(1/\epsilon))$
  - accelerated gradient method: $\mathcal{O}(\sqrt{\kappa} \log(1/\epsilon))$
  - stochastic gradient method: $\mathcal{O}(\kappa/\epsilon)$ (population loss)

Distributed gradient methods

distributed implementation of gradient descent

$$\underset{w \in \mathbf{R}^d}{\text{minimize}} \quad P(w) = \frac{1}{N} \sum_{i=1}^{m} \Phi_i(X_{i:} w)$$



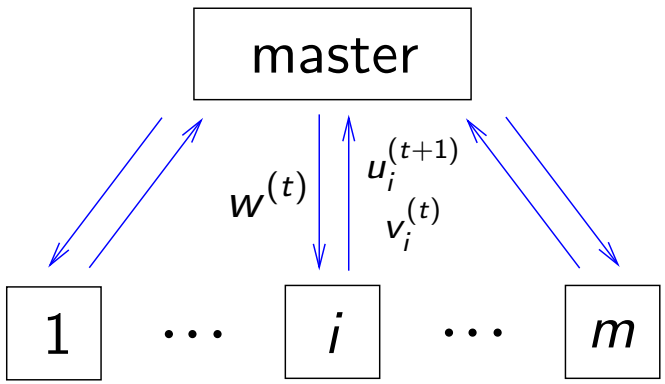
- each iteration involves one round of communication
- number of communication rounds: $\mathcal{O}(\kappa \log(1/\epsilon))$
- can use accelerated gradient method: $\mathcal{O}(\sqrt{\kappa} \log(1/\epsilon))$

ADMM

- reformulation: $\begin{array}{ll} \text{minimize} & \frac{1}{N}\sum_{i=1}^m f_i(u_i) \\ \text{subject to} & u_i = w, \quad i = 1, \ldots, m \end{array}$

- augmented Lagrangian

$$L_\rho(u, v, w) = \sum_{i=1}^m \left(f_i(u_i) + \langle v_i, u_i - w \rangle + \frac{\rho}{2}\|u_i - w\|_2^2 \right)$$



$w^{(t+1)} = \arg\min_w L_\rho(u^{(t+1)}, v^{(t)}, w)$

communicate $\mathcal{O}(d)$ bits

$u_i^{(t+1)} = \arg\min_{u_i} L_\rho(u_i, v^{(t)}, w^{(t)})$

$v_i^{(t+1)} = v_i^{(t)} + \rho(u_i^{(t)} - w^{(t)})$

- no. of communication rounds: $\mathcal{O}(\kappa \log(1/\epsilon))$ or $\mathcal{O}(\sqrt{\kappa} \log(1/\epsilon))$

The dual ERM problem

primal problem

$$\underset{w \in \mathbf{R}^d}{\text{minimize}} \;\; P(w) \stackrel{\text{def}}{=} \frac{1}{N} \sum_{i=1}^{m} \Phi_i(X_{i:} w) + g(w)$$

dual problem

$$\underset{\alpha \in \mathbf{R}^N}{\text{maximize}} \;\; D(\alpha) \stackrel{\text{def}}{=} -\frac{1}{N} \sum_{i=1}^{m} \Phi_i^*(\alpha_i) - g^* \left(-\frac{1}{N} \sum_{i=1}^{m} (X_{i:})^T \alpha_i \right)$$

where g^* and ϕ_i^* are convex conjugate functions

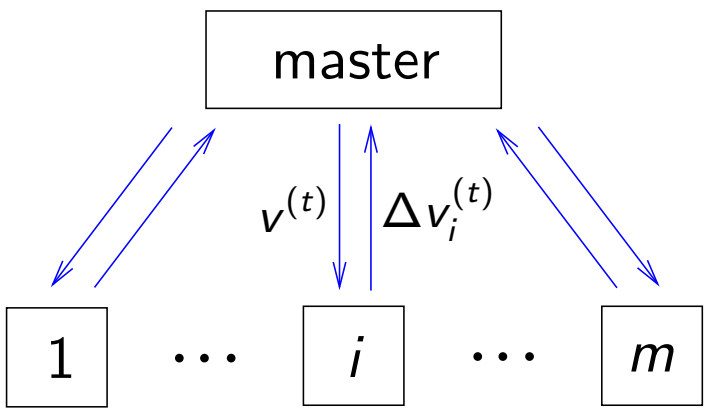
- $g^*(v) = \sup_{u \in \mathbf{R}^d} \{v^T u - g(u)\}$
- $\Phi_i^*(\alpha_i) = \sup_{z \in \mathbf{R}^{n_i}} \{\alpha_i^T z - \Phi_i(z)\}$, for $i = 1, \ldots, m$

recover primal variable from dual: $w = \nabla g^* \left(-\frac{1}{N} \sum_{i=1}^{m} (X_{i:})^T \alpha_i \right)$

The CoCoA(+) algorithm

(Jaggi et al. 2014, Ma et al. 2015)

$$\underset{\alpha\in\mathbf{R}^N}{\text{maximize}}\ D(\alpha) \overset{\text{def}}{=} -\frac{1}{N}\sum_{i=1}^{m}\Phi_i^*(\alpha_i) - g^*\left(-\frac{1}{N}\sum_{i=1}^{m}(X_{i:})^T\alpha_i\right)$$



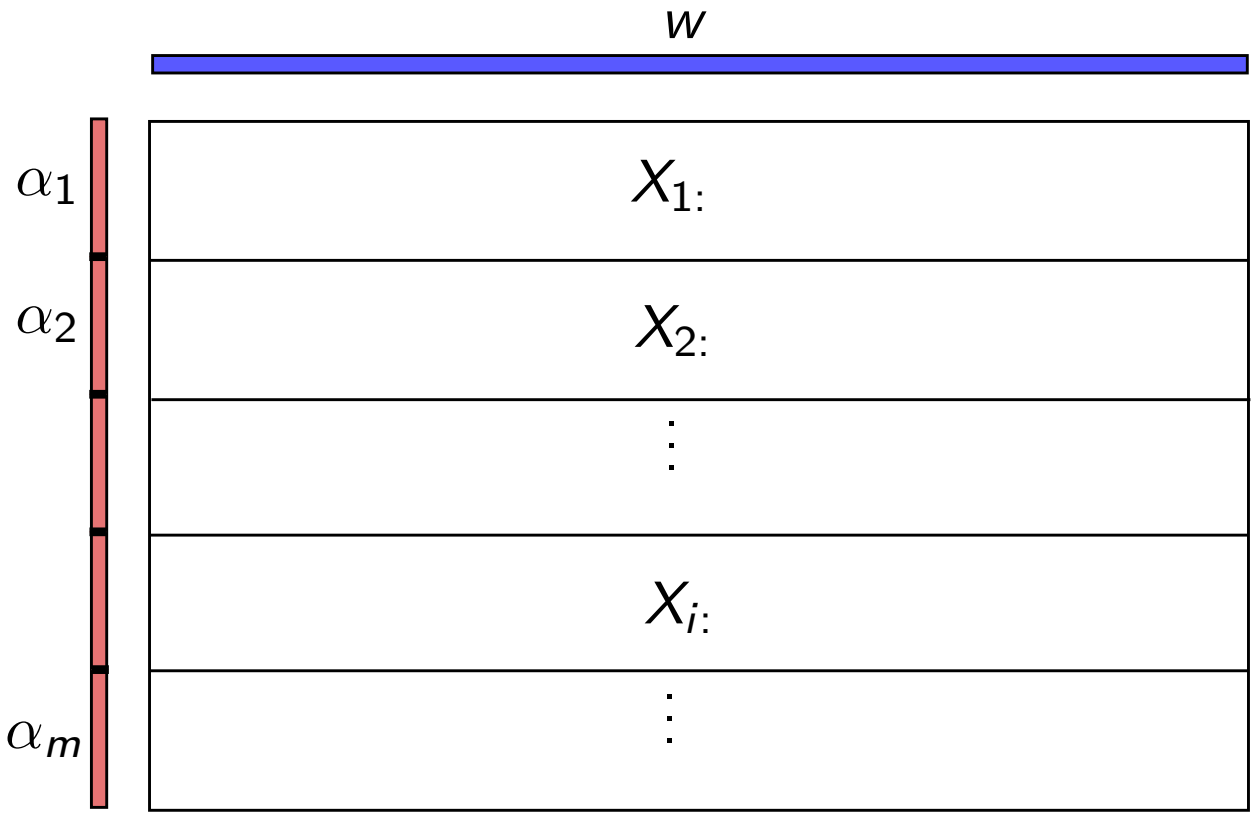
$$v^{(t+1)} = v^{(t)} + \sum_{i=1}^{m}\Delta v_i^{(t)}$$

communicate $\mathcal{O}(d)$ bits

$$\alpha_i^{(t+1)} = \arg\max_{\alpha_i}\mathcal{G}_i(v^{(t)},\alpha_i)$$
$$\Delta v_i^{(t)} = \tfrac{1}{N}(X_{i:})^T(\alpha_i^{(t+1)} - \alpha_i^{(t)})$$

- each iteration involves one round of communication
- number of communication rounds: $\mathcal{O}(\kappa\log(1/\epsilon))$
- can be accelerated by PPA (Catalyst, Lin et al.): $\mathcal{O}(\sqrt{\kappa}\log(1/\epsilon))$

Primal and dual variables



$$w = \nabla g^* \left(-\frac{1}{N} \sum_{i=1}^{m} (X_{i:})^T \alpha_i \right)$$

Can we do better?

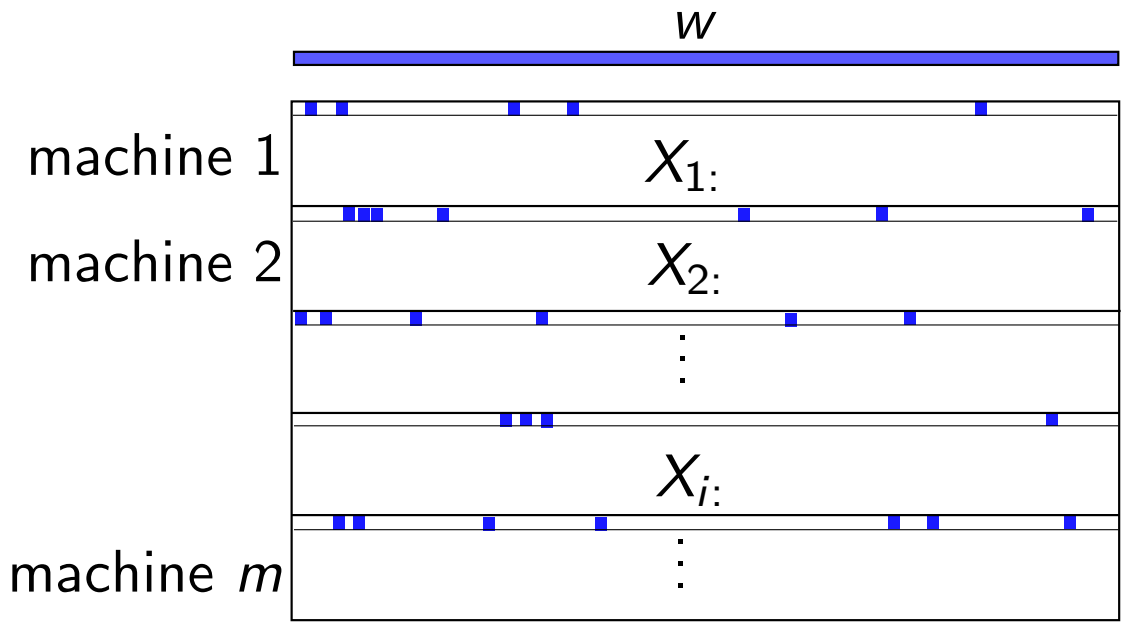
- asynchronous distributed algorithms?
- better communication complexity?
- better computation complexity?

Outline

- distributed empirical risk minimization
- **randomized primal-dual algorithms with parameter servers**
- variance reduction techniques
- DSCOVR algorithms
  (Doubly Stochastic Coordinate Optimization with Variance Reduction)
- preliminary experiments

Asynchronism: Hogwild! style

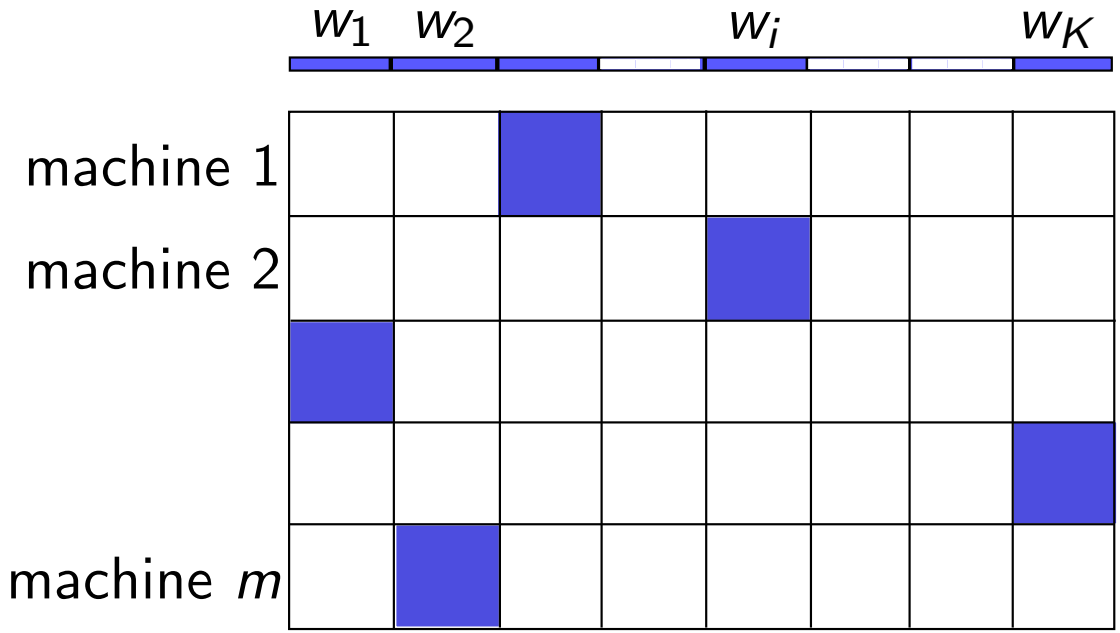
idea: exploit sparsity to avoid simultaneous updates (Niu et al. 2011)



problems:

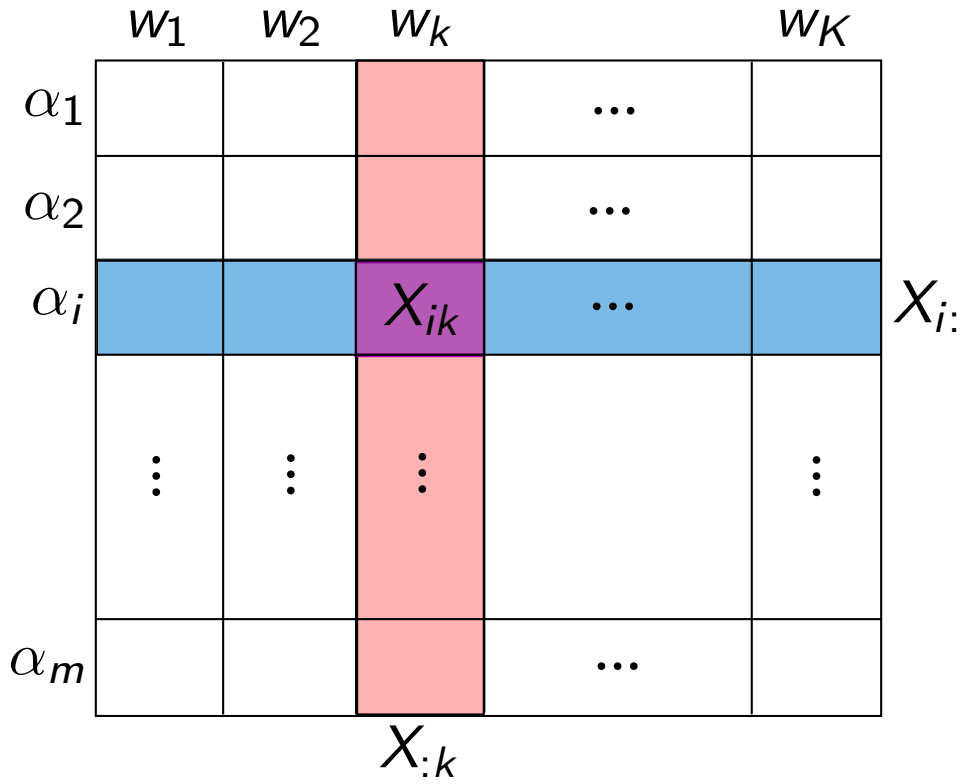
- too frequent communication (bottleneck for distributed system)
- slow convergence (sublinear rate using stochastic gradients)

Tame the hog: forced separation



- partition w into K blocks $w_1, \ldots, w_K$
- each machine updates a different block using relevant columns
- set $K > m$ so that all machines can work all the time
- event-driven asynchronism:
  - whenever free, each machine request new block to update
  - update orders can be intentionally randomized

Double separation via saddle-point formulation



$$\min_{w \in \mathbf{R}^d} \max_{\alpha \in \mathbf{R}^N} \left\{ \frac{1}{N} \sum_{i=1}^{m} \sum_{k=1}^{K} \alpha_i^T X_{ik} w_k - \frac{1}{N} \sum_{i=1}^{m} \Phi_i^*(\alpha_i) + \sum_{k=1}^{K} g(w_k) \right\}$$

A randomized primal-dual algorithm

Algorithm 1: Doubly stochastic primal-dual coordinate update

input: initial points $w^{(0)}$ and $\alpha^{(0)}$
for $t = 0, 1, 2, \ldots, T-1$

1. pick $j \in \{1, \ldots, m\}$ and $l \in \{1, \ldots, K\}$ with probabilities p_j and q_l
2. compute stochastic gradients

$$u_j^{(t+1)} = \tfrac{1}{q_l} X_{jl} w_l^{(t)}, \qquad v_l^{(t+1)} = \tfrac{1}{p_j} \tfrac{1}{N} (X_{jl})^T \alpha_j^{(t)}$$

3. update primal and dual block coordinates:

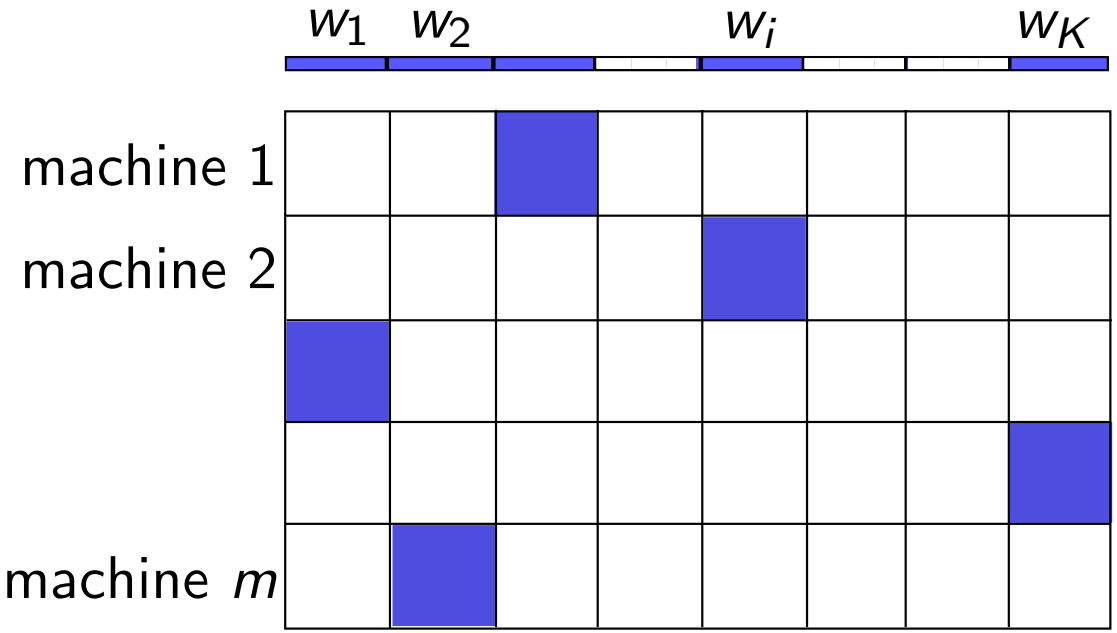
$$\alpha_i^{(t+1)} = \begin{cases} \operatorname{prox}_{\sigma_j \psi_j^*}\left(\alpha_j^{(t)} + \sigma_j u_j^{(t+1)}\right) & \text{if } i = j, \\ \alpha_i^{(t)}, & \text{if } i \neq j, \end{cases}$$

$$w_k^{(t+1)} = \begin{cases} \operatorname{prox}_{\tau_l g_l}\left(w_l^{(t)} - \tau_l v_l^{(t+1)}\right) & \text{if } k = l, \\ w_k^{(t)}, & \text{if } k \neq l. \end{cases}$$

end for

How good is this algorithm?

- on the update order
  - sequence $(i(t), k(t))$ not really i.i.d.
  - in practice better than i.i.d.?



- bad news: sublinear convergence, with complexity $O(1/\epsilon)$

Outline

- distributed empirical risk minimization
- randomized primal-dual algorithms with parameter servers
- **variance reduction techniques**
- DSCOVR algorithms
  (Doubly Stochastic Coordinate Optimization with Variance Reduction)
- preliminary experiments

Minimizing finite average of convex functions

$$\text{minimize} \quad F(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w) + g(w)$$

- **batch proximal gradient method**

$$w^{(t+1)} = \text{prox}_{\eta_t g}\left(w^{(t)} - \eta_t \nabla F(w^{(t)})\right)$$

  - each step very expensive, relatively fast convergence
  - can use accelerated proximal gradient methods

- **stochastic proximal gradient method**

$$w^{(t+1)} = \text{prox}_{\eta_t g}\left(w^{(t)} - \eta_t \nabla f_{i_t}(w^{(t)})\right) \qquad (i_t \text{ chosen randomly})$$

  - each iteration very cheap, but very slow convergence
  - accelerated stochastic algorithms do not really help

- recent advances in **randomized algorithms**:
  exploit finite average (sum) structure to get best of both worlds

Stochastic variance reduced gradient (SVRG)

- SVRG (Johnson & Zhang 2013)
  - update form

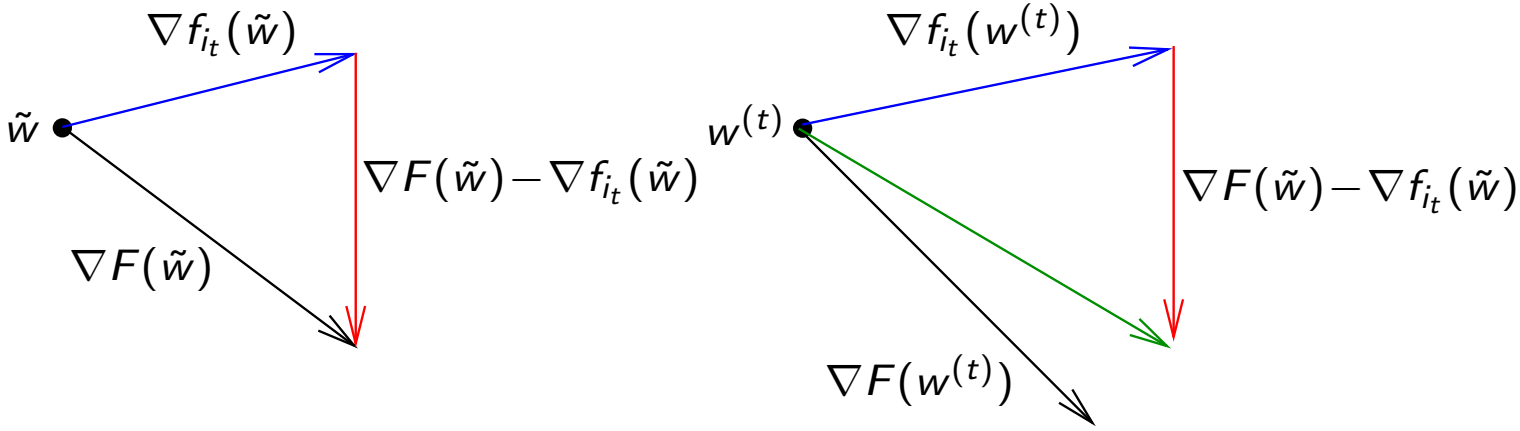
$$w^{(t+1)} = w^{(t)} - \eta(\nabla f_{i_t}(w^{(t)}) - \nabla f_{i_t}(\tilde{w}) + \nabla F(\tilde{w}))$$

  - update $\tilde{w}$ periodically (every few passes)
- still a stochastic gradient method

$$\mathbf{E}_{i_t}[\nabla f_{i_t}(w^{(t)}) - \nabla f_{i_t}(\tilde{w}) + \nabla F(\tilde{w})] = \nabla F(w^{(t)})$$

  - expected update direction is the same as $\mathbf{E}[\nabla f_{i_t}(w^{(t)})]$
  - variance can be diminishing if $\tilde{w}$ updated periodically
- complexity: $O\left((n+\kappa)\log\frac{1}{\epsilon}\right)$, cf. SGD $O(\kappa/\epsilon)$
- Prox-SVRG (X. and Zhang 2014): same complexity

Intuition of variance reduction



SAGA (Defazio, Bach & Lacoste-Julien 2014)

- the algorithm

$$w^{(t+1)} = w^{(t)} - \eta_t \left[\nabla f_{i_t}(w^{(t)}) - \nabla f_{i_t}(z_{i_t}^{(t)}) + \frac{1}{n} \sum_{j=1}^{n} \nabla f_j(z_j^{(t)}) \right]$$

  $z_j^{(t)}$: last point at which component gradient ∇f_j was calculated

- naturally extends to proximal version
- complexity: $O\left((n+\kappa)\log\frac{1}{\epsilon}\right)$, cf. SGD $O(\kappa/\epsilon)$

Condition number and batch complexity

- **condition number:** $\kappa = \frac{R^2}{\lambda\gamma}$ (considering $\kappa \gg 1$)
- **batch complexity:** number of equivalent passes over dataset

complexities to reach $\mathbf{E}[P(w^{(t)}) - P^\star] \leq \epsilon$

algorithm	iteration complexity	batch complexity
stochastic gradient	$(1+\kappa)/\epsilon$	$(1+\kappa)/(n\epsilon)$
full gradient (FG)	$(1+\kappa')\log(1/\epsilon)$	$(1+\kappa')\log(1/\epsilon)$
accelerated FG (Nesterov)	$(1+\sqrt{\kappa'})\log(1/\epsilon)$	$(1+\sqrt{\kappa'})\log(1/\epsilon)$
SDCA, SAG(A), SVRG, ...	$(n+\kappa)\log(1/\epsilon)$	$(1+\kappa/n)\log(1/\epsilon)$
A-SDCA, APCG, SPDC, ...	$(n+\sqrt{\kappa n})\log(1/\epsilon)$	$(1+\sqrt{\kappa/n})\log(1/\epsilon)$

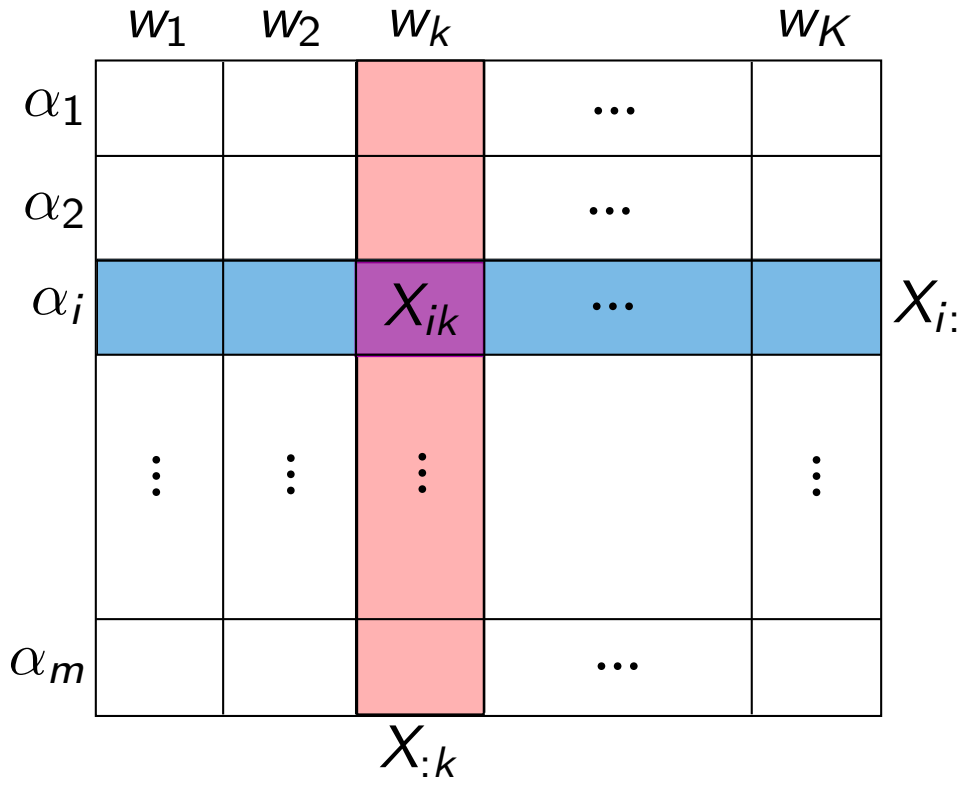
SDCA:	Shalev-Shwartz & Zhang (2013)	SAGA:	Defazio, Bach & Lacoste-Julien (2014)
SAG:	Schmidt, Le Roux, & Bach (2012, 2013)	A-SDCA:	Shalev-Shwartz & Zhang (2014)
Finito:	Defazio, Caetano & Domke (2014)	MISO:	Mairal (2015)
SVRG:	Johnson & Zhang (2013), X. & Zhang (2014)	APCG:	Lin, Lu & X. (2014)
Quartz:	Qu, Richtárik, & Zhang (2015)	SPDC:	Zhang & X. (2015)
Catalyst:	Lin, Mairal, & Harchaoui (2015)	A-APPA	Frostig, Ge, Kakade, &Sidford (2015)
RPDG:	Lan (2015)		**and others ...**

lower bound: Agarwal & Bottou (2015), Lan (2015), Woodworth & Srebro (2016)

Outline

- distributed empirical risk minimization
- randomized primal-dual algorithms with parameter servers
- variance reduction techniques
- **DSCOVR algorithms**
  (Doubly Stochastic Coordinate Optimization with Variance Reduction)
- preliminary experiments

Double separation via saddle-point formulation



$$\min_{w \in \mathbf{R}^d} \max_{\alpha \in \mathbf{R}^N} \left\{ \frac{1}{N} \sum_{i=1}^{m} \sum_{k=1}^{K} \alpha_i^T X_{ik} w_k - \frac{1}{N} \sum_{i=1}^{m} \Phi_i^*(\alpha_i) + \sum_{k=1}^{K} g(w_k) \right\}$$

Algorithm 2: DSCOVR-SVRG

for $s = 0, 1, 2, \ldots, S-1$

- $\bar{u}^{(s)} = X\bar{w}^{(s)}$ and $\bar{v}^{(s)} = \frac{1}{N} X^T \bar{\alpha}^{(s)}$
- $w^{(0)} = \bar{w}^{(s)}$ and $\alpha^{(0)} = \bar{\alpha}^{(s)}$
- **for** $t = 0, 1, 2, \ldots, T-1$

  1. pick $j \in \{1, \ldots, m\}$ and $l \in \{1, \ldots, K\}$ with probabilities p_j and q_l
  2. compute variance-reduced stochastic gradients:

$$u_j^{(t+1)} = \bar{u}_j^{(s)} + \frac{1}{q_l} X_{jl}\left(w_l^{(t)} - \bar{w}_l^{(s)}\right), \quad v_l^{(t+1)} = \bar{v}_l^{(s)} + \frac{1}{p_j}\frac{1}{N}(X_{jl})^T\left(\alpha_j^{(t)} - \bar{\alpha}_j^{(s)}\right)$$

  3. update primal and dual block coordinates:

$$\alpha_i^{(t+1)} = \begin{cases} \operatorname{prox}_{\sigma_j \Psi_j^*}\left(\alpha_j^{(t)} + \sigma_j u_j^{(t+1)}\right) & \text{if } i = j, \\ \alpha_i^{(t)}, & \text{if } i \neq j, \end{cases}$$

$$w_k^{(t+1)} = \begin{cases} \operatorname{prox}_{\tau_l g_l}\left(w_l^{(t)} - \tau_l v_l^{(t+1)}\right) & \text{if } k = l, \\ w_k^{(t)}, & \text{if } k \neq l. \end{cases}$$

  end for

- $\bar{w}^{(s+1)} = w^{(T)}$ and $\bar{\alpha}^{(s+1)} = \alpha^{(T)}$.

end for

Convergence analysis of DSCOVR-SVRG

- **assumptions:**
  - each ϕ_i is $1/\gamma$-smooth $\Longrightarrow$ ϕ_i^* is γ-strongly convex

$$|\phi_i'(\alpha) - \phi_i'(\beta)| \leq (1/\gamma)|\alpha - \beta|, \quad \forall\, \alpha, \beta \in \mathbf{R}$$

  - g is λ-strongly convex $\Longrightarrow$ g^* is $1/\lambda$-smooth

$$g(w) \geq g(u) + g'(u)^T(w-u) + \frac{\lambda}{2}\|w-u\|_2^2, \quad \forall\, w, u \in \mathbf{R}^d$$

- **strong duality**
  - there exist unique $(w^\star, \alpha^\star)$ satisfying $P(w^\star) = D(\alpha^\star)$
  - $w^\star = \nabla g^*\big(-\frac{1}{N}\sum_{i=1}^m (X_{i:})^T \alpha_i^\star\big)$

Theorem: Let Λ and Γ be two constants that satisfy

$$\Lambda \;\geq\; \|X_{ik}\|^2, \quad \text{for all} \quad i = 1, \ldots, m, \text{ and } j = 1, \ldots, K,$$

$$\Gamma \;\geq\; \max_{i,k} \left\{ \frac{1}{p_i} \left(1 + \frac{9m\Lambda}{2q_k N \lambda \gamma} \right), \; \frac{1}{q_k} \left(1 + \frac{9K\Lambda}{2p_i N \lambda \gamma} \right) \right\}.$$

If we choose the step sizes as

$$\begin{aligned} \sigma_i &= \frac{1}{2\gamma(p_i\Gamma - 1)}, \qquad i = 1, \ldots, m, \\ \tau_k &= \frac{1}{2\lambda(q_k\Gamma - 1)}, \qquad k = 1, \ldots, K, \end{aligned}$$

and the number of iterations during each stage $T \geq \log(3)\Gamma$, then

$$\mathbf{E}\left[\left\| \begin{array}{c} \bar{w}^{(s)} - w^\star \\ \bar{\alpha}^{(s)} - \alpha^\star \end{array} \right\|^2_{\lambda, \frac{\gamma}{N}} \right] \leq \left(\frac{2}{3} \right)^s \left\| \begin{array}{c} \bar{w}^{(0)} - w^\star \\ \bar{\alpha}^{(0)} - \alpha^\star \end{array} \right\|^2_{\lambda, \frac{\gamma}{N}}$$

Complexity analysis (assuming $K > m$)

- if $p_i = \frac{1}{m}$ and $q_k = \frac{1}{K}$, then can take $\Gamma = K\left(1 + \frac{9mK\Lambda}{2N\lambda\gamma}\right)$
- if $p_i = \frac{\|X_{i:}\|_F^2}{\|X\|_F^2}$ and $q_k = \frac{\|X_{:k}\|_F^2}{\|X\|_F^2}$, then $\Gamma = K\left(1 + \frac{9\|X\|_F^2}{2N\lambda\gamma}\right)$
- if $\max_i \|x_i\| \leq R$, then can use $\Gamma = K\left(1 + \frac{9R^2}{2\lambda\gamma}\right) = K\left(1 + \frac{9}{2}\kappa\right)$

complexities

- iteration complexity (number of X_{ik} blocks processed):

$$O\left(K(1 + m + \kappa)\log\tfrac{1}{\epsilon}\right)$$

- communication complexity (number of d-vectors transmitted):

$$O\left((1 + m + \kappa)\log\tfrac{1}{\epsilon}\right)$$

- computation complexity (number of passes over whole dataset):

$$O\left((1 + \tfrac{\kappa}{m})\log\tfrac{1}{\epsilon}\right)$$

Convergence of duality gap

Theorem: Let Λ and Γ be two constants that satisfy

$$\Lambda \;\geq\; \|X_{ik}\|^2, \quad \text{for all} \quad i = 1, \ldots, m, \text{ and } j = 1, \ldots, K,$$

$$\Gamma \;\geq\; \max_{i,k} \left\{ \frac{1}{p_i} \left(1 + \frac{18 m \Lambda}{q_k N \lambda \gamma} \right), \; \frac{1}{q_k} \left(1 + \frac{18 K \Lambda}{p_i N \lambda \gamma} \right) \right\}.$$

If we choose the step sizes as

$$\begin{aligned} \sigma_i &= \frac{1}{\gamma(p_i \Gamma - 1)}, \qquad i = 1, \ldots, m, \\ \tau_k &= \frac{1}{\lambda(q_k \Gamma - 1)}, \qquad k = 1, \ldots, K, \end{aligned}$$

and the number of iterations during each stage $T \geq \log(3)\Gamma$, then

$$\mathbf{E}\left[P(\bar{w}^{(s)}) - D(\bar{\alpha}^{(s)}) \right] \;\leq\; \left(\frac{2}{3} \right)^s 3\Gamma \left(P(\bar{w}^{(0)}) - D(\bar{\alpha}^{(0)}) \right).$$

Algorithm 3: DSCOVR-SAGA

- $\bar{u}^{(0)} = Xw^{(0)}$ and $\bar{v}^{(0)} = \frac{1}{N} X^T \alpha^{(0)}$
- **for** $t = 0, 1, 2, \ldots, T-1$

  1. pick $i \in \{1, \ldots, m\}$ and $k \in \{1, \ldots, K\}$ with probabilities p_i and q_k
  2. compute variance-reduced stochastic gradients:

$$\begin{aligned} u_i^{(t+1)} &= \bar{u}_i^{(t)} - \tfrac{1}{q_k} U_{ik}^{(t)} + \tfrac{1}{q_k} X_{ik} w_k^{(t)} \\ v_k^{(t+1)} &= \bar{v}_k^{(t)} - \tfrac{1}{p_i} (V_{ik}^{(t)})^T + \tfrac{1}{p_i} \tfrac{1}{N} (X_{ik})^T \alpha_i^{(t)} \end{aligned}$$

  3. update primal and dual block coordinates:

$$\begin{aligned} \alpha_i^{(t+1)} &= \operatorname{prox}_{\sigma_i \Psi_i^*} \left(\alpha_i^{(t)} + \sigma_j u_i^{(t+1)} \right) \\ w_k^{(t+1)} &= \operatorname{prox}_{\tau_k g_k} \left(w_k^{(t)} - \tau_k v_k^{(t+1)} \right) \end{aligned}$$

  4. update averaged historical stochastic gradients:

$$\bar{u}_i^{(t+1)} = \bar{u}_i^{(t)} - U_{ik}^{(t)} + X_{ik} w_k^{(t)}, \quad \bar{v}_k^{(t+1)} = \bar{v}_k^{(t)} - (V_{ik}^{(t)})^T + \tfrac{1}{N} (X_{ik})^T \alpha_i^{(t)}$$

  5. update the table of historical stochastic gradients:

$$U_{ik}^{(t+1)} = X_{ik} w_k^{(t)}, \qquad V_{ik}^{(t+1)} = \tfrac{1}{N} \left((X_{ik})^T \alpha_i^{(t)} \right)^T$$

end for

Convergence of DSCOVR-SAGA

Theorem: Let Λ and Γ be two constants that satisfy

$$\Lambda \;\geq\; \|X_{ik}\|^2, \quad i = 1, \ldots, m, \quad j = 1, \ldots, K,$$

$$\Gamma \;\geq\; \max_{i,k} \left\{ \frac{1}{p_i} \left(1 + \frac{9m\Lambda}{2q_k N \lambda \gamma} \right),\; \frac{1}{q_k} \left(1 + \frac{9K\Lambda}{2p_i N \lambda \gamma} \right),\; \frac{1}{p_i q_k} \right\}.$$

If we choose the step sizes as

$$\begin{aligned} \sigma_i &= \frac{1}{2\gamma(p_i\Gamma - 1)}, \qquad i = 1, \ldots, m, \\ \tau_k &= \frac{1}{2\lambda(q_k\Gamma - 1)}, \qquad k = 1, \ldots, K, \end{aligned}$$

then for $t = 1, 2, \ldots,$

$$\mathbf{E}\left[\left\| \begin{array}{c} w^{(t)} - w^\star \\ \alpha^{(t)} - \alpha^\star \end{array} \right\|^2_{\lambda, \frac{\gamma}{N}} \right] \;\leq\; \left(1 - \frac{1}{3\Gamma} \right)^t \frac{4}{3} \left\| \begin{array}{c} w^{(0)} - w^\star \\ \alpha^{(0)} - \alpha^\star \end{array} \right\|^2_{\lambda, \frac{\gamma}{N}}$$

Algorithm 4: Accelerated DSCOVR

input: initial points $\widetilde{w}^{(0)}, \widetilde{\alpha}^{(0)}$, and parameter $\delta > 0$

for $r = 0, 1, 2, \ldots,$

1. find an approximate saddle point of

$$L_\delta^{(r)}(w, a) = L(w, \alpha) + \frac{\delta\lambda}{2}\|w - \widetilde{w}^{(r)}\|^2 - \frac{\delta\gamma}{2N}\|\alpha - \widetilde{\alpha}^{(r)}\|^2$$

   using one of the following two options:

   - *option 1:* let $S = \frac{2\log(2(1+\delta))}{\log(3/2)}$ and $T = \log(3)\Gamma_\delta$, and

$$(\widetilde{w}^{(r+1)}, \widetilde{\alpha}^{(r+1)}) = \mathrm{DSCOVR\text{-}SVRG}(\widetilde{w}^{(r)}, \widetilde{\alpha}^{(r)}, S, T)$$

   - *option 2:* let $T = 6\log\left(\frac{8(1+\delta)}{3}\right)\Gamma_\delta$ and

$$(\widetilde{w}^{(r+1)}, \widetilde{\alpha}^{(r+1)}) = \mathrm{DSCOVR\text{-}SAGA}(\widetilde{w}^{(r)}, \widetilde{\alpha}^{(r)}, T)$$

end for

(following techniques in Balamurugan and Bach 2016)

Convergence of accelerated DSCOVR

Theorem: Let Λ and Γ_δ be two constants that satisfy

$$\Lambda \;\geq\; \|X_{ik}\|^2, \quad \text{for all} \quad i = 1, \ldots, m, \text{ and } j = 1, \ldots, K,$$

$$\Gamma_\delta \;\geq\; \max_{i,k} \left\{ \frac{1}{p_i} \left(1 + \frac{9m\Lambda}{2q_k N \lambda \gamma (1+\delta)^2} \right), \; \frac{1}{q_k} \left(1 + \frac{9K\Lambda}{2p_i N \lambda \gamma (1+\delta)^2} \right) \right\}.$$

If we choose the step sizes as

$$\begin{aligned} \sigma_i &= \frac{1}{2\gamma(p_i \Gamma_\delta - 1)}, \qquad i = 1, \ldots, m, \\ \tau_k &= \frac{1}{2\lambda(q_k \Gamma_\delta - 1)}, \qquad k = 1, \ldots, K, \end{aligned}$$

then

$$\mathbf{E}\left[\left\| \begin{array}{c} \widetilde{w}^{(r)} - w^\star \\ \widetilde{\alpha}^{(r)} - \alpha^\star \end{array} \right\|^2_{\lambda, \frac{\gamma}{N}} \right] \;\leq\; \left(1 - \frac{1}{2(1+\delta)} \right)^{2r} \left\| \begin{array}{c} \widetilde{w}^{(0)} - w^\star \\ \widetilde{\alpha}^{(0)} - \alpha^\star \end{array} \right\|^2_{\lambda, \frac{\gamma}{N}}$$

Complexity of accelerated DSCOVR

- simplified expression for the constant $\Gamma_\delta = K\left(1 + \frac{9\kappa}{2(1+\delta)^2}\right)$
- total number of block updates

$$O\left(K\left(m(1+\delta) + \frac{9\kappa}{2(1+\delta)}\right)\log(1+\delta)\log\left(\frac{1}{\epsilon}\right)\right).$$

if we choose $\delta = \sqrt{9\kappa/(2m)} - 1$ (assuming $\kappa > m$), then

$$O\left(K\sqrt{m\kappa}\log(1+\delta)\log\left(\frac{1}{\epsilon}\right)\right).$$

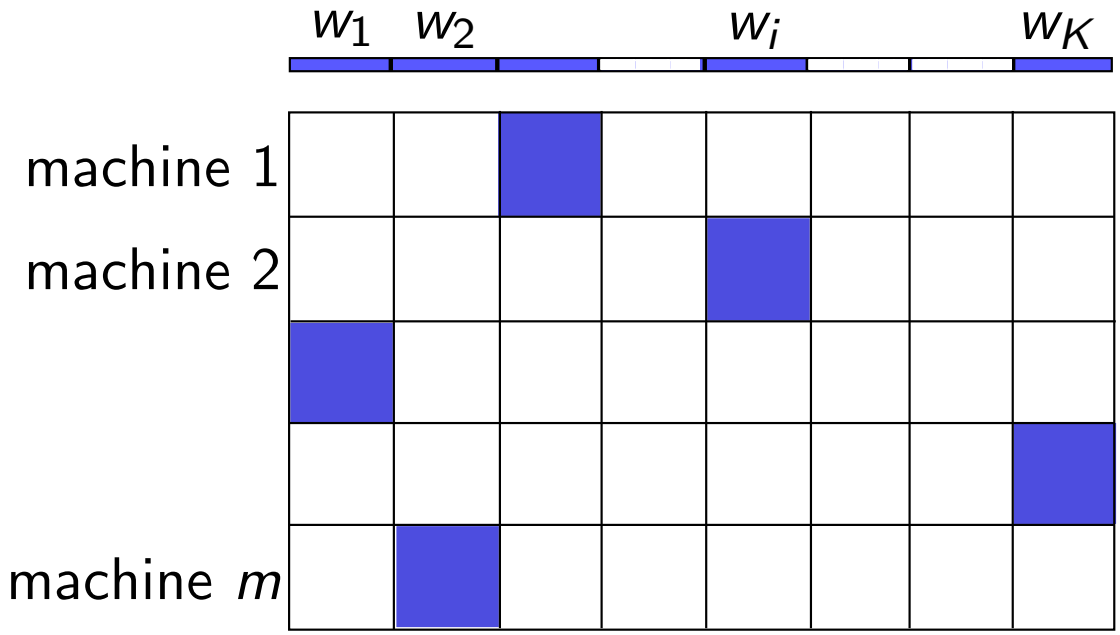
- communication complexity (number of d-vectors transmitted):

$$O\left(\sqrt{m\kappa}\log\frac{1}{\epsilon}\right)$$

- computation complexity (number of passes over whole dataset):

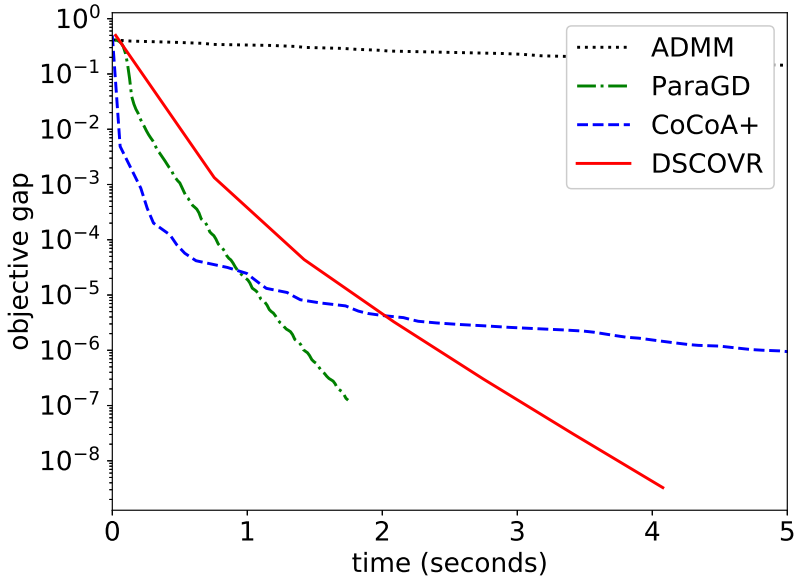
$$O\left(\left(1 + \sqrt{\frac{\kappa}{m}}\right)\log\frac{1}{\epsilon}\right)$$

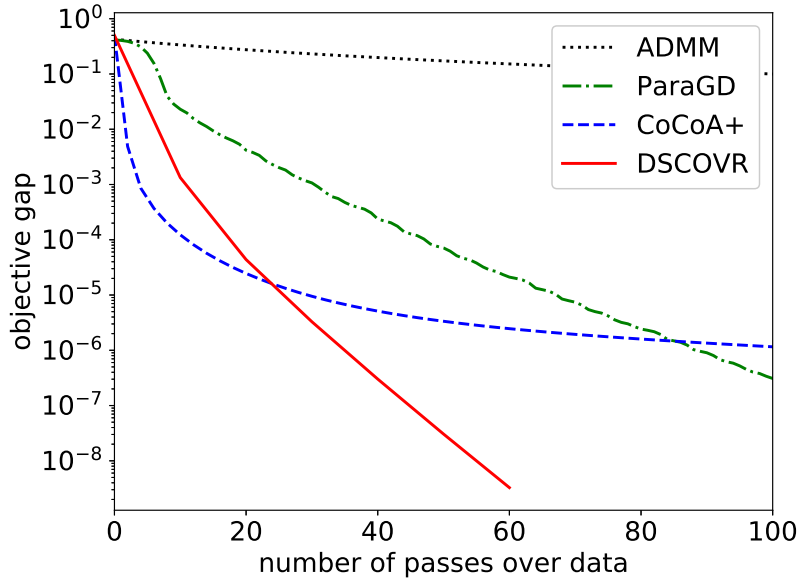
Implementation of DSCOVR



- C++, efficient sparse matrix operations using OpenMP
- asynchronous implementatino: MPI nonblocking Send/IRecv
- also implemented Parallel GD, ADMM, CoCoA(+)
- more to come ...

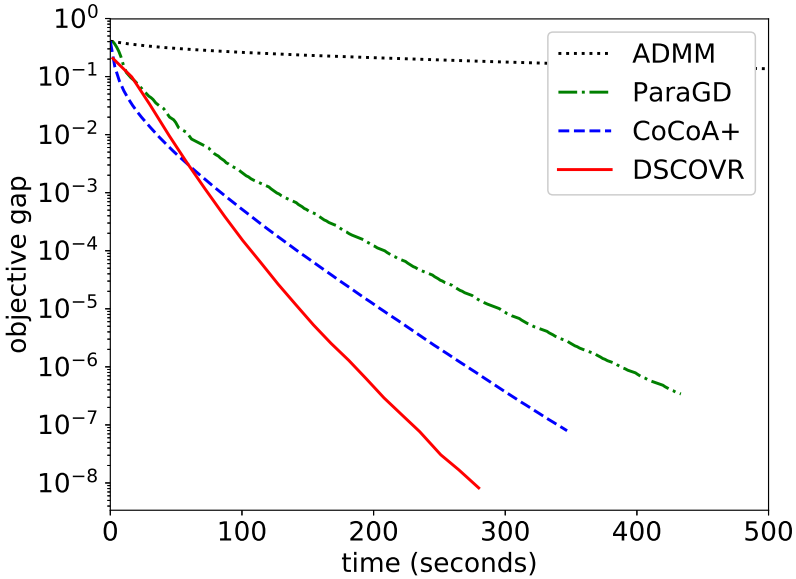
Experiments with RCV1.binary dataset

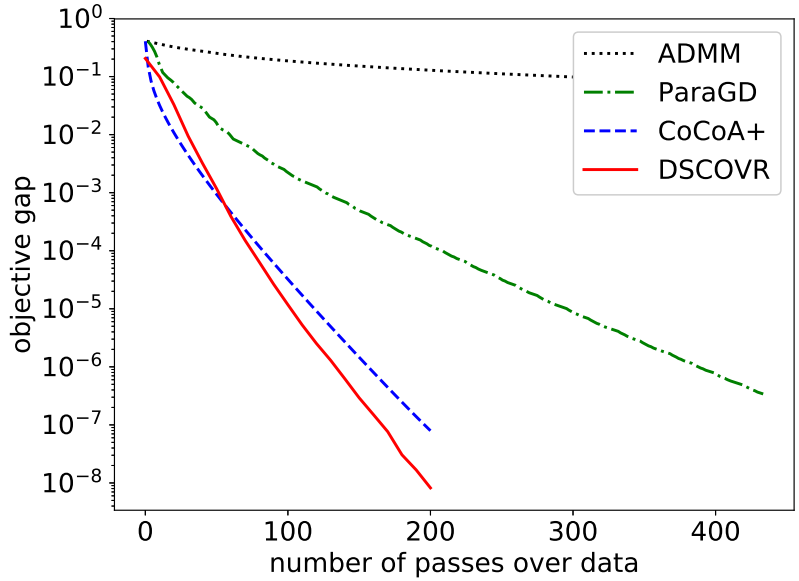




- $N = 677,399$, $d = 47236$, row normalizd with $R = 1$
- run on cluster of 20 machines, 5 parameter servers, 1 master
- randomly shuffled sample and features
- smoothed hinge loss with ℓ_2 regularization, $\lambda = 10^{-4}$

Experiments with webspam dataset



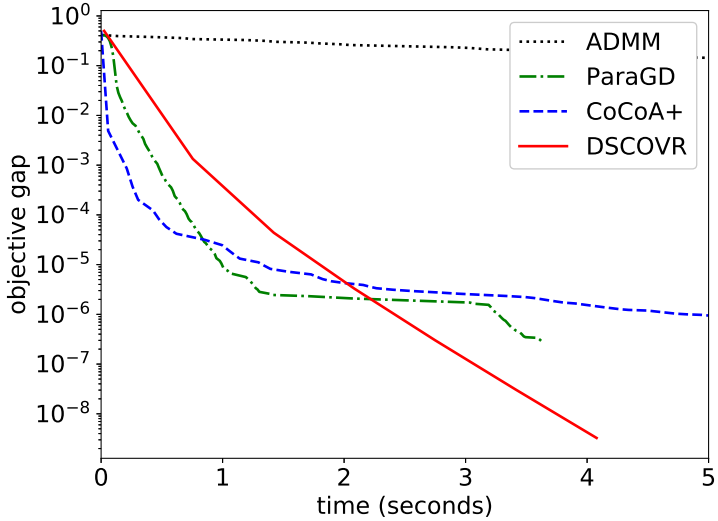


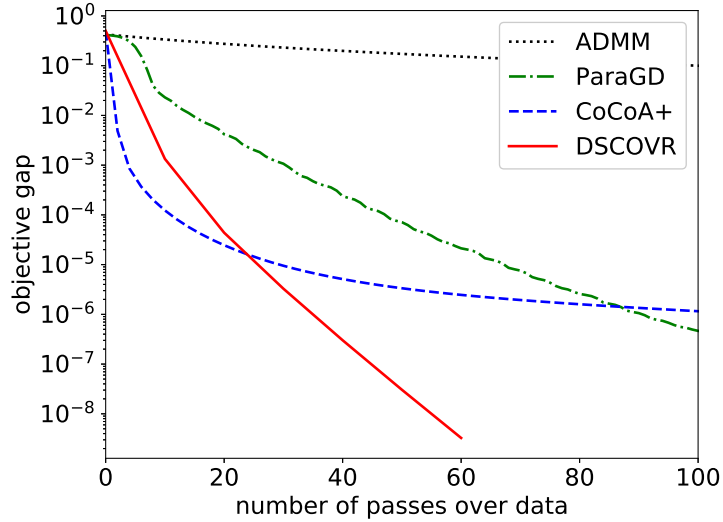
- $N = 350,000$, $d = 16,609,143$, row normalizd with $R = 1$
- run on cluster of 20 workers, 10 parameter servers, 1 master
- randomly shuffled sample and features
- logistic regression with ℓ_2 regularization, $\lambda = 10^{-4}$

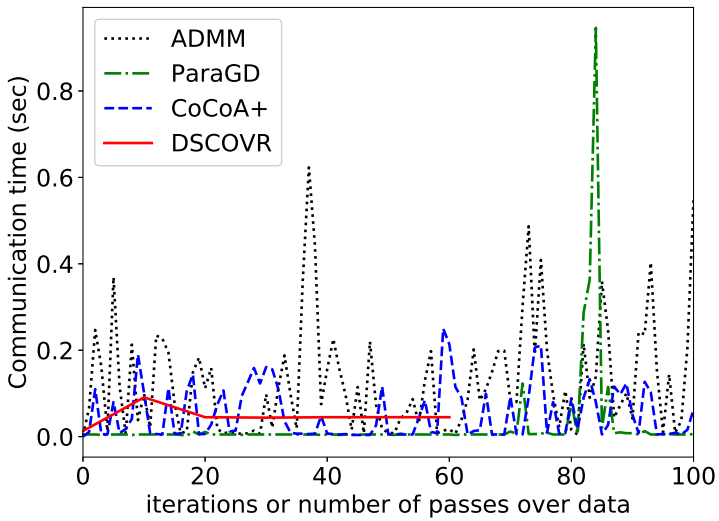
DSCOVR-SAGA on webspam dataset

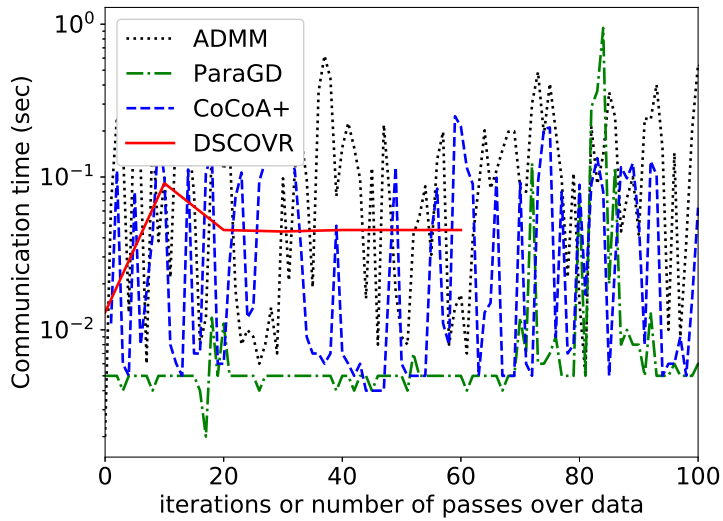
nSync	nEpoch	primal_obj	dual_obj	gap	t_sync	t_comp	t_comm	t_loop	t_elpsd
0	0	0.430232537706	0.225168873757	2.051e-01	2.025	0.487	1.453	2.025	2.025
1	10	0.361465626442	0.262779737691	9.869e-02	2.127	6.435	7.831	14.266	16.291
2	20	0.311349950700	0.278401966087	3.295e-02	2.050	5.685	8.062	13.747	30.037
3	30	0.294032397911	0.284556248547	9.476e-03	2.096	6.058	8.788	14.845	44.882
4	40	0.289940053505	0.286701605120	3.238e-03	2.024	5.422	8.101	13.524	58.406
5	50	0.288706980240	0.287536154538	1.171e-03	2.044	5.367	8.095	13.470	71.877
6	60	0.288254740784	0.287864269333	3.905e-04	2.035	6.212	8.790	14.993	86.870
7	70	0.288128681323	0.287978130088	1.506e-04	2.004	5.569	8.110	13.680	100.550
8	80	0.288088497819	0.288025094667	6.340e-05	2.031	5.436	8.097	13.532	114.081
9	90	0.288073396692	0.288046902858	2.649e-05	2.024	5.364	8.049	13.422	127.503
10	100	0.288068226887	0.288056477572	1.175e-05	2.030	5.364	8.068	13.421	140.925
11	110	0.288066217652	0.288060941805	5.276e-06	2.030	5.336	8.037	13.378	154.303
12	120	0.288065430239	0.288062901758	2.528e-06	2.030	5.334	8.108	13.437	167.740
13	130	0.288065194360	0.288063899046	1.295e-06	2.024	5.337	8.028	13.364	181.104
14	140	0.288065015129	0.288064394949	6.202e-07	2.029	5.318	8.064	13.403	194.507
15	150	0.288064917447	0.288064625062	2.924e-07	2.026	5.353	8.003	13.357	207.864
16	160	0.288064885386	0.288064735092	1.503e-07	2.030	5.387	8.073	13.439	221.302
17	170	0.288064867950	0.288064791393	7.656e-08	2.039	5.625	8.078	13.704	235.006
18	180	0.288064852335	0.288064821789	3.055e-08	2.023	6.698	9.328	16.023	251.029
19	190	0.288064850799	0.288064834053	1.675e-08	2.031	5.736	8.052	13.790	264.820
20	200	0.288064848282	0.288064840064	8.218e-09	2.003	6.378	8.633	15.025	279.845

The cost of synchronization









Summary

DSCOVR

- saddle-point formulation allows simultaneous partition of both data and model to gain parallelism
- used stochastic variance reduction to achieve fast convergence
- asynchronous, event-driven implementation
- no simulataneous updates, no stale states of delays to worry
- improved computation complexity for distributed ERM

additional features

- DSCOVR-SVRG only need to communicate sparse vectors
- also developed dual-free version of primal-dual algorithms (using technique from Lan 2015)