DiSCO: Communication-Efficient Distributed Optimization of Self-Concordant Empirical Loss

Yuchen Zhang
UC Berkeley

Joint work with
Lin Xiao (Microsoft Research)

July, 2015

Motivation

big data optimization problems

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

origins

- data science (statistics, machine learning, data mining, . . .)
- industry (search, online services, advertising, social media, . . .)

tools:

- regression/classification
- numerical linear algebra
- statistical estimation

Empirical risk minimization (ERM)

- popular formulation in supervised learning

$$\underset{w \in \mathbf{R}^d}{\text{minimize}} \quad \frac{1}{N} \sum_{j=1}^{N} \phi(w, z_j) + \frac{\lambda}{2} \|w\|_2^2$$

  - i.i.d. data samples: $z_1, \ldots, z_N \in \mathcal{Z} \subset \mathbf{R}^p$
  - loss function: ϕ convex in w for every $z \in \mathcal{Z}$
  - regularization parameter $\lambda \sim 1/\sqrt{N}$ or smaller

Empirical risk minimization (ERM)

- popular formulation in supervised learning

$$\underset{w \in \mathbf{R}^d}{\text{minimize}} \quad \frac{1}{N} \sum_{j=1}^{N} \phi(w, z_j) + \frac{\lambda}{2} \|w\|_2^2$$

  - i.i.d. data samples: $z_1, \ldots, z_N \in \mathcal{Z} \subset \mathbf{R}^p$
  - loss function: ϕ convex in w for every $z \in \mathcal{Z}$
  - regularization parameter $\lambda \sim 1/\sqrt{N}$ or smaller

- **linear regression**
  - quadratic loss.

- **binary classification:**
  - logistic loss.
  - hinge loss.

Distributed optimization

- **distributed algorithms:** alternate between
  - a local computation procedure at each machine
  - a communication round with simple map-reduce operations
- **bottleneck:** high cost of inter-machine communication
  - speed/delay, synchronization
  - energy consumption
- **iteration complexity**
  - number of communication rounds to find $f(\widehat{w}) - f(w_\star) \leq \epsilon$
  - measures computation and communication efficiency

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 (GD): $\mathcal{O}(\kappa \log(1/\epsilon))$
  - accelerated GD, ADMM: $\mathcal{O}(\sqrt{\kappa} \log(1/\epsilon))$
  - stochastic gradient method (SGD): $\mathcal{O}(\kappa/\epsilon)$

Implications for distributed machine learning

$$f(w) = \frac{1}{m}\sum_{i=1}^{m} f_i(w) = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} \phi(w, z_{i,j}) + \frac{\lambda}{2}\|w\|_2^2$$

- m machines, each holds n samples. Regularization parameter λ decreases with total sample size mn.
  (typical setting: $\lambda \sim \frac{1}{\sqrt{mn}}$, condition number $\kappa = \Theta(\sqrt{mn})$)

- iteration complexity of accelerated GD or ADMM:

$$\mathcal{O}\left((mn)^{1/4}\log(1/\epsilon)\right)$$

  iteration complexity grows with sample size!

Implications for distributed machine learning

$$f(w) = \frac{1}{m}\sum_{i=1}^{m} f_i(w) = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} \phi(w, z_{i,j}) + \frac{\lambda}{2}\|w\|_2^2$$

- m machines, each holds n samples. Regularization parameter λ decreases with total sample size mn.
  (typical setting: $\lambda \sim \frac{1}{\sqrt{mn}}$, condition number $\kappa = \Theta(\sqrt{mn})$)

- iteration complexity of accelerated GD or ADMM:

$$\mathcal{O}\left((mn)^{1/4}\log(1/\epsilon)\right)$$

  iteration complexity grows with sample size!

- can we do better?
  - unlikely for first-order methods (optimal complexity)
  - need to use further structure and/or alternative methods

  our approach: Newton + preconditioned conjugate gradient

Making it really work

- **outer loop:** (*inexact*) damped Newton method
  - problem: iteration complexity $\mathcal{O}\big(\kappa^2 + \log(\log(1/\epsilon))\big)$
  - solution: self-concordant analysis
    $$O\big(f(w_0) - f(w_\star) + \log(\log(1/\epsilon))\big)$$
- **inner loop:** conjugate gradient (CG) to compute Newton step
  - problem: iteration complexity of CG: $\mathcal{O}(\sqrt{\kappa}\log(1/\epsilon)$
  - solution: use preconditioned conjugate gradient (PCG) method

Making it really work

- **outer loop:** (*inexact*) damped Newton method
  - problem: iteration complexity $\mathcal{O}\big(\kappa^2 + \log(\log(1/\epsilon))\big)$
  - solution: self-concordant analysis
    $$O\big(f(w_0) - f(w_\star) + \log(\log(1/\epsilon))\big)$$
- **inner loop:** conjugate gradient (CG) to compute Newton step
  - problem: iteration complexity of CG: $\mathcal{O}(\sqrt{\kappa}\log(1/\epsilon)$
  - solution: use preconditioned conjugate gradient (PCG) method
- **DiSCO:** what's in a name?
  - Distributed SeCond Order method
  - Distributed Self-Concordant Optimization
  - Distributed Stochastic Convex Optimization

Preview: communication efficiency

Algorithm	Number of Communication Rounds $\widetilde{\mathcal{O}}(\cdot)$	
	Ridge Regression (quadratic loss)	Binary Classification (logistic loss, smoothed hinge loss)
AGD	$(mn)^{1/4}\log(1/\epsilon)$	$(mn)^{1/4}\log(1/\epsilon)$
ADMM	$(mn)^{1/4}\log(1/\epsilon)$	$(mn)^{1/4}\log(1/\epsilon)$
DANE	$m\log(1/\epsilon)$	$(mn)^{1/2}\log(1/\epsilon)$
DiSCO	$m^{1/4}\log(1/\epsilon)$	$m^{3/4}d^{1/4} + m^{1/4}d^{1/4}\log(1/\epsilon)$

- regularization parameter $\lambda \sim 1/\sqrt{mn}$
- deterministic or high probability bounds (except last one)
- DANE by Shamir, Srebro and Zhang (ICML 2014)

DiSCO: Inexact damped Newton method

input: initial point w_0 and specification of a sequence $\{\epsilon_k\}$

repeat for $k = 0, 1, 2, \ldots$

1. find a vector v_k such that $\|f''(w_k)v_k - f'(w_k)\|_2 \leq \epsilon_k$
2. compute $\delta_k = \sqrt{v_k^T f''(w_k) v_k}$ and $w_{k+1} = w_k - \frac{1}{1+\delta_k} v_k$

until a stopping criterion is satisfied

- if $\delta_k = 0$ and $\epsilon_k = 0$, reduces to classical Newton method.
- forcing $\delta_k > 0$: ensuring convergence when w_k is far apart from the optimal solution.
- allowing $\epsilon_k > 0$: v_k can be computed by a communication efficient distributed algorithm (preconditioned CG).

Iteration complexity

- **assumptions:**
  - f standard self-concordant
  - $\lambda I \preceq f''(w) \preceq LI$ for all $w \in \mathbf{R}^d$
- **theorem:** if we choose

$$\epsilon_k \sim \|f'(w_k)\|_2$$

  then $f(w_k) - f(w_\star) \leq \epsilon$ whenever

$$k \approx f(w_0) - f(w_\star) + \log(1/\epsilon)$$

- **theorem:** if initialize w_0 by one-shot averaging, then $f(w_0) - f(w_\star) = O(1)$.

Solving Newton system

- need to find v_k such that $\|f''(w_k)v_k - f'(w_k)\|_2 \leq \epsilon_k$.

Solving Newton system

- need to find v_k such that $\|f''(w_k)v_k - f'(w_k)\|_2 \leq \epsilon_k$.
- **conjugate gradient** (CG) method to approximately solve $f''(w_k)v_k = f'(w_k)$: iteration complexity $\sqrt{\kappa}$, where $\kappa = L/\lambda \sim \sqrt{mn}$ is the condition number of Hessian $f''(w_k)$.

Solving Newton system

- need to find v_k such that $\|f''(w_k)v_k - f'(w_k)\|_2 \leq \epsilon_k$.
- **conjugate gradient** (CG) method to approximately solve $f''(w_k)v_k = f'(w_k)$: iteration complexity $\sqrt{\kappa}$, where $\kappa = L/\lambda \sim \sqrt{mn}$ is the condition number of Hessian $f''(w_k)$.
- **precondition conjugate gradient** (PCG) method to approximately solve $P^{-1}f''(w_k)v_k = P^{-1}f'(w_k)$:
  - Let μ be a scalar and $H_1 := f_1''(w_k)$ be the local Hessian at the first machine; then Let $P := H_1 + \mu I$.

Solving Newton system

- need to find v_k such that $\|f''(w_k)v_k - f'(w_k)\|_2 \leq \epsilon_k$.
- **conjugate gradient** (CG) method to approximately solve $f''(w_k)v_k = f'(w_k)$: iteration complexity $\sqrt{\kappa}$, where $\kappa = L/\lambda \sim \sqrt{mn}$ is the condition number of Hessian $f''(w_k)$.
- **precondition conjugate gradient** (PCG) method to approximately solve $P^{-1}f''(w_k)v_k = P^{-1}f'(w_k)$:
  - Let μ be a scalar and $H_1 := f_1''(w_k)$ be the local Hessian at the first machine; then Let $P := H_1 + \mu I$.
  - **theorem**: if $\|H_1 - f''(w_k)\|_2 \leq \mu$, then the condition number of $P^{-1}f''(w_k)$ is bounded by $1 + \mu/\lambda$.

Solving Newton system

- need to find v_k such that $\|f''(w_k)v_k - f'(w_k)\|_2 \leq \epsilon_k$.
- **conjugate gradient** (CG) method to approximately solve $f''(w_k)v_k = f'(w_k)$: iteration complexity $\sqrt{\kappa}$, where $\kappa = L/\lambda \sim \sqrt{mn}$ is the condition number of Hessian $f''(w_k)$.
- **precondition conjugate gradient** (PCG) method to approximately solve $P^{-1}f''(w_k)v_k = P^{-1}f'(w_k)$:
  - Let μ be a scalar and $H_1 := f_1''(w_k)$ be the local Hessian at the first machine; then Let $P := H_1 + \mu I$.
  - **theorem**: if $\|H_1 - f''(w_k)\|_2 \leq \mu$, then the condition number of $P^{-1}f''(w_k)$ is bounded by $1 + \mu/\lambda$.
  - **theorem**: by matrix concentration, $\mu \sim 1/\sqrt{n}$, thus the condition number of $P^{-1}f''(w_k)$ is bounded by $\sqrt{m}$.

Solving Newton system

- need to find v_k such that $\|f''(w_k)v_k - f'(w_k)\|_2 \leq \epsilon_k$.
- **conjugate gradient** (CG) method to approximately solve $f''(w_k)v_k = f'(w_k)$: iteration complexity $\sqrt{\kappa}$, where $\kappa = L/\lambda \sim \sqrt{mn}$ is the condition number of Hessian $f''(w_k)$.
- **precondition conjugate gradient** (PCG) method to approximately solve $P^{-1}f''(w_k)v_k = P^{-1}f'(w_k)$:
  - Let μ be a scalar and $H_1 := f_1''(w_k)$ be the local Hessian at the first machine; then Let $P := H_1 + \mu I$.
  - **theorem**: if $\|H_1 - f''(w_k)\|_2 \leq \mu$, then the condition number of $P^{-1}f''(w_k)$ is bounded by $1 + \mu/\lambda$.
  - **theorem**: by matrix concentration, $\mu \sim 1/\sqrt{n}$, thus the condition number of $P^{-1}f''(w_k)$ is bounded by $\sqrt{m}$.
  - PCG's iteration complexity ($m^{1/4}$) is much smaller than CG's iteration complexity ($(mn)^{1/4}$).

Distributed PCG method

input: $w_k \in \mathbf{R}^d$ and $\mu \geq 0$,

communicate: broadcast w_k, receive $f_i'(w_k)$, and form $f'(w_k)$

initialize: $v^{(0)} = 0,\ s^{(0)} = P^{-1}r^{(0)},\ r^{(0)} = f'(w_k),\ u^{(0)} = s^{(0)}$

repeat for $t = 0, 1, 2 \ldots,$

1. compute $Hu^{(t)} := \sum_{i=1}^m f_i''(w_k)u^{(t)}$ by map-reduce.
2. compute $\alpha_t = \frac{\langle r^{(t)}, s^{(t)} \rangle}{\langle u^{(t)}, Hu^{(t)} \rangle}$ and update:
$$v^{(t+1)} = v^{(t)} + \alpha_t u^{(t)}, \qquad r^{(t+1)} = r^{(t)} - \alpha_t H u^{(t)}$$
3. compute $\beta_t = \frac{\langle r^{(t+1)}, s^{(t+1)} \rangle}{\langle r^{(t)}, s^{(t)} \rangle}$ and update:
$$s^{(t+1)} = P^{-1}r^{(t+1)}, \qquad u^{(t+1)} = s^{(t+1)} + \beta_t u^{(t)}$$

until $\|r^{(t+1)}\|_2 \leq \epsilon_k$

return: $v_k = v^{(t+1)}$ and $\delta_k = \sqrt{v_k^T H v^{(t)} + \alpha^{(t)} v_k^T H u^{(t)}}$

Putting pieces together

- **outer loop (inexact damped Newton method):** iteration complexity $\sim 1 + \log(1/\epsilon)$. (if data i.i.d. and $\lambda \sim 1/\sqrt{mn}$)
- **inner loop (PCG):** iteration complexity $\sim m^{1/4}$.
  (if data i.i.d. and $\lambda \sim 1/\sqrt{mn}$)
- **overall iteration complexity**: $\sim m^{1/4} \log(1/\epsilon)$.
- only communicate first-order information in each round.
- applies to quadratic loss, logistic loss, smoothed hinge loss and any other self-concordant loss functions.

Experiments

regularized logistic regression

$$\ell(w) = \frac{1}{N}\sum_{i=1}^{N} \log(1+\exp(-y_i(w^T x_i))) + \frac{\gamma}{2}\|w\|_2^2$$

- data normalized so that $\|x_i\| = 1$ for all $i = 1, \ldots, N$
- regularization parameter set to be $\gamma = 10^{-5}$

Dataset name	sample size N	dimension d	sparsity
Covtype ($N \gg d$)	581,012	54	22%
RCV1 ($N \approx d$)	20,242	47,236	0.16%
News20 ($N \ll d$)	19,996	1,355,191	0.04%

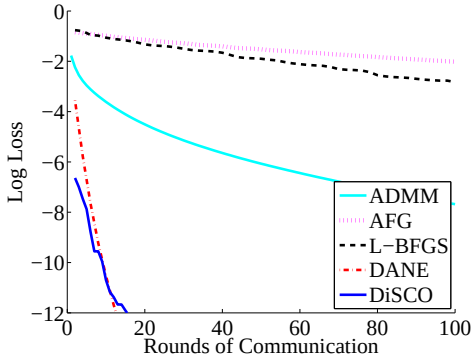
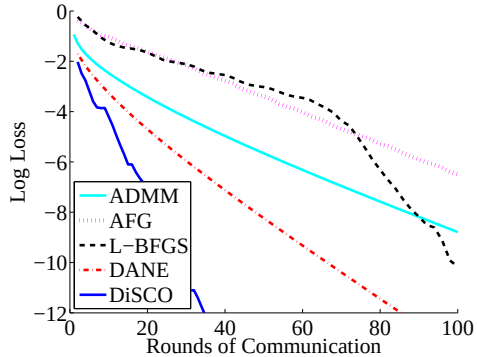
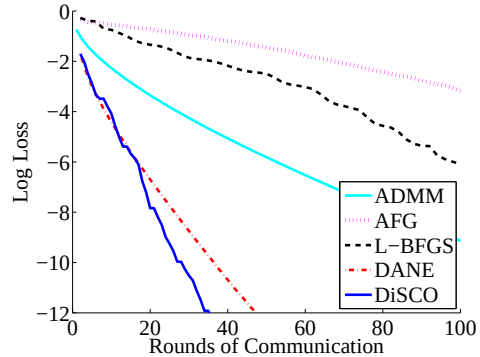
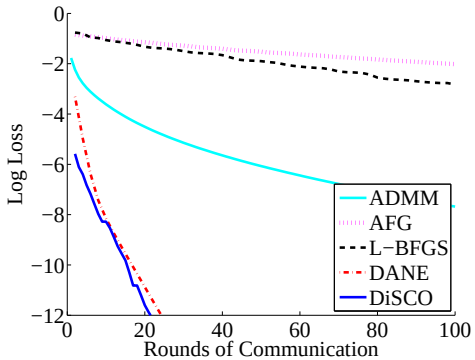
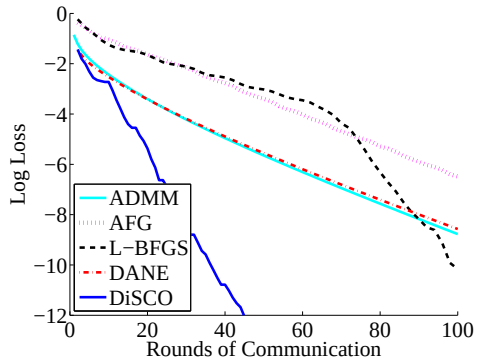
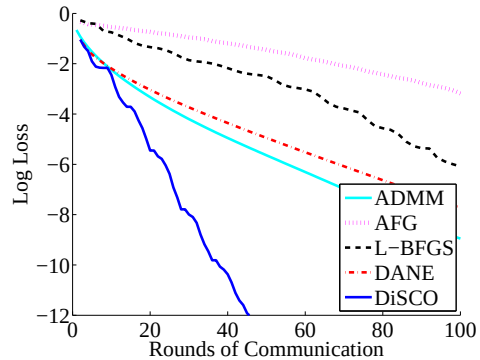
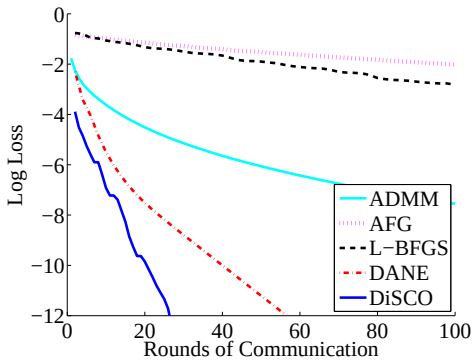
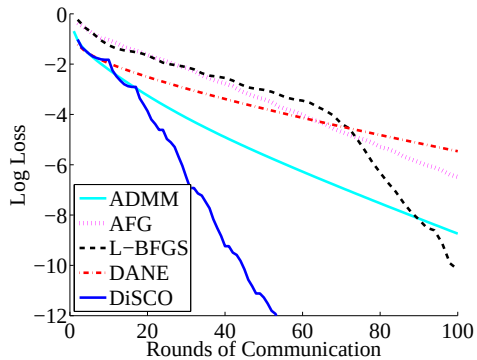
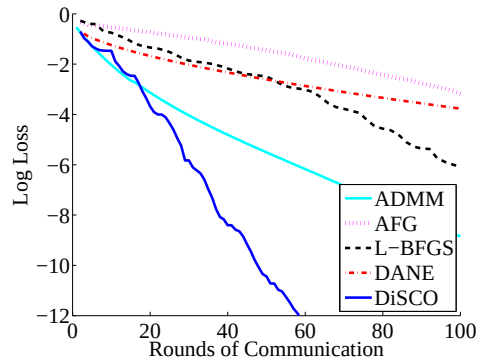
Algorithms compared

- AFG: distributed implementation of accelerated full gradient method, with adaptive line search to speed up convergence
- ADMM: manually tuned penalty parameter for best performance
- L-BFGS: memory size 30
- DANE: manually tuned parameter μ for best performance

$$v_{k+1,i} = \arg\min_{w \in \mathbf{R}^d} \left\{ f_i(w) - \langle f_i'(w_k) - f'(w_k), w \rangle + \frac{\mu}{2} \|w - w_k\|_2^2 \right\}$$

$$w_{k+1} = (1/m) \sum_{i=1}^{m} v_{k+1,i}$$

- DiSCO: better performance without self-concordant scaling

m	Covtype	RCV1	News20
4			
16			
64			

Conclusions

DiSCO

- a communication-efficient distributed optimization method
- computation and communication rounds
  - does not grow with sample size and condition number
  - only grows slowly with number of machines

Conclusions

DiSCO

- a communication-efficient distributed optimization method
- computation and communication rounds
  - does not grow with sample size and condition number
  - only grows slowly with number of machines

- algorithm
  - inexact damped Newton method
  - preconditioned conjugate gradient method
- utilizes
  - i.i.d. property of the data
  - local second-order information