Recent Advances in Primal-Dual Coordinate Methods for Empirical Risk Minimization

Qi Lei *

Based on joint work with

Ian E.H. Yen†, Chao-yuan Wu*, Inderjit S. Dhillon* and Pradeep Ravikumar†.

* University of Texas at Austin
† Carnegie Mellon University

Problem Setup

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}) &= \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x}) \\ &\overset{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} \phi_i(\boldsymbol{a}_i^\top \boldsymbol{x}) + g(\boldsymbol{x}) \end{aligned}$$

- Large-scale supervised machine learning: very large feature dimension d, and very large data size n
- Assumptions: μ-strongly convex, L-smooth. Condition number $\kappa = \frac{L}{\mu}$.

Problem Setup

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x})$$
$$\stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} \phi_i(\boldsymbol{a}_i^\top \boldsymbol{x}) + g(\boldsymbol{x})$$

- Large-scale supervised machine learning: very large feature dimension d, and very large data size n
- Assumptions: μ-strongly convex, L-smooth. Condition number $\kappa = \frac{L}{\mu}$.

  e.g.,
  squared loss $\phi \Rightarrow$ linear regression problem
  sigmoid loss $\phi \Rightarrow$ logistic regression problem
  smooth hinge loss $\phi \Rightarrow$ support vector machine

First-Order Optimization

$$\begin{aligned}\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}) &= \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x}) \\ &\overset{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} \phi_i(\boldsymbol{a}_i^\top \boldsymbol{x}) + g(\boldsymbol{x})\end{aligned}$$

- Classical methods comparison

$\min_{\boldsymbol{x}} f(\boldsymbol{x})$	method	per Iteration	#Iteration
	GD	$\mathcal{O}(nd)$	$\mathcal{O}(\kappa \log^{-1}\epsilon)$
	RCD	$\mathcal{O}(n)$	$\mathcal{O}(d\kappa \log^{-1}\epsilon)$
	SGD	$\mathcal{O}(d)$	$\mathcal{O}(\kappa\epsilon^{-1})$

- Expansions

method	per Iteration	# Iteration
GD	$\mathcal{O}(nd)$	$\mathcal{O}(\kappa \log^{-1} \epsilon)$
RCD	$\mathcal{O}(n)$	$\mathcal{O}(d\kappa \log^{-1} \epsilon)$
SGD	$\mathcal{O}(d)$	$\mathcal{O}(\kappa\epsilon^{-1})$
MISO	$\mathcal{O}(d)$	$\mathcal{O}((n+\kappa) \log^{-1} \epsilon)$
SVRG		
SAG		$\mathcal{O}((n+\sqrt{n\kappa}) \log^{-1} \epsilon)$
SAGA		if accelerated
SDCA	$\mathcal{O}(d)$	$\mathcal{O}((n+\kappa) \log^{-1} \epsilon)$
SPDC	$\mathcal{O}(d)$	$\mathcal{O}((n+\sqrt{n\kappa}) \log^{-1} \epsilon)$

- Expansions

method	per Iteration	# Iteration
GD	$\mathcal{O}(nd)$	$\mathcal{O}(\kappa \log^{-1} \epsilon)$
RCD	$\mathcal{O}(n)$	$\mathcal{O}(d\kappa \log^{-1} \epsilon)$
SGD	$\mathcal{O}(d)$	$\mathcal{O}(\kappa\epsilon^{-1})$
MISO	$\mathcal{O}(d)$	$\mathcal{O}((n+\kappa) \log^{-1} \epsilon)$
SVRG		
SAG		$\mathcal{O}((n+\sqrt{n\kappa}) \log^{-1} \epsilon)$
SAGA		if accelerated
SDCA	$\mathcal{O}(d)$	$\mathcal{O}((n+\kappa) \log^{-1} \epsilon)$
SPDC	$\mathcal{O}(d)$	$\mathcal{O}((n+\sqrt{n\kappa}) \log^{-1} \epsilon)$

from lazy gradient evaluations to variance reduction:
MISO (Mairal, 2015)
SVRG (Johnson & Zhang, 2013)
SAG(A) (Defazio et al., 2014)

- Expansions

method	per Iteration	# Iteration
GD	$\mathcal{O}(nd)$	$\mathcal{O}(\kappa \log^{-1} \epsilon)$
RCD	$\mathcal{O}(n)$	$\mathcal{O}(d\kappa \log^{-1} \epsilon)$
SGD	$\mathcal{O}(d)$	$\mathcal{O}(\kappa\epsilon^{-1})$
MISO	$\mathcal{O}(d)$	$\mathcal{O}((n+\kappa) \log^{-1} \epsilon)$
SVRG		
SAG		$\mathcal{O}((n+\sqrt{n\kappa}) \log^{-1} \epsilon)$
SAGA		if accelerated
SDCA	$\mathcal{O}(d)$	$\mathcal{O}((n+\kappa) \log^{-1} \epsilon)$
SPDC	$\mathcal{O}(d)$	$\mathcal{O}((n+\sqrt{n\kappa}) \log^{-1} \epsilon)$

similar update rule but improved convergence rate
SDCA (Shalev-Shwartz and Zhang, 2013)

- Expansions

method	per Iteration	# Iteration
GD	$\mathcal{O}(nd)$	$\mathcal{O}(\kappa \log^{-1} \epsilon)$
RCD	$\mathcal{O}(n)$	$\mathcal{O}(d\kappa \log^{-1} \epsilon)$
SGD	$\mathcal{O}(d)$	$\mathcal{O}(\kappa\epsilon^{-1})$
MISO	$\mathcal{O}(d)$	$\mathcal{O}((n+\kappa) \log^{-1} \epsilon)$
SVRG		
SAG		$\mathcal{O}((n+\sqrt{n\kappa}) \log^{-1} \epsilon)$
SAGA		if accelerated
SDCA	$\mathcal{O}(d)$	$\mathcal{O}((n+\kappa) \log^{-1} \epsilon)$
SPDC	$\mathcal{O}(d)$	$\mathcal{O}((n+\sqrt{n\kappa}) \log^{-1} \epsilon)$

added extrapolation terms
SPDC (Zhang and Xiao, 2015)

Primal, Dual, Lagrangian forms

Primal form

$$\min_{\boldsymbol{x}\in\mathbb{R}^d}\left\{P(\boldsymbol{x})\stackrel{\text{def}}{=}\frac{1}{n}\sum_{i=1}^{n}\phi_i(\boldsymbol{a}_i^\top\boldsymbol{x})+g(\boldsymbol{x})\right\}$$

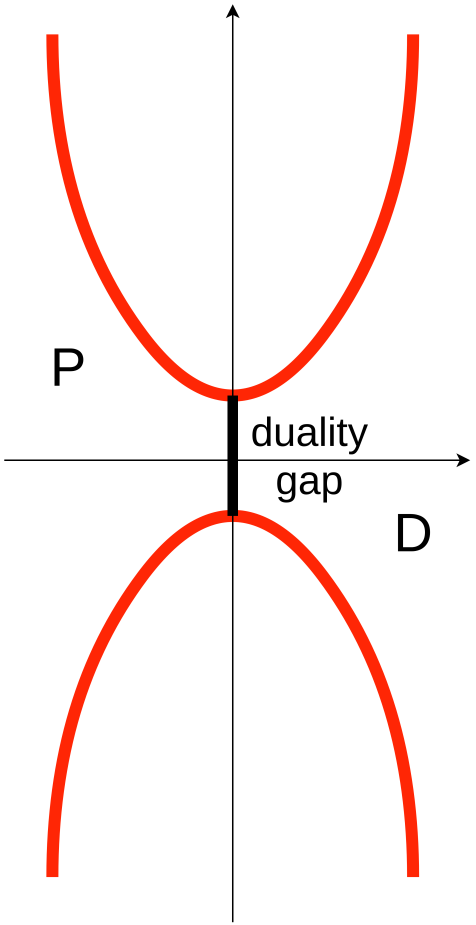
$\Rightarrow$ Dual form

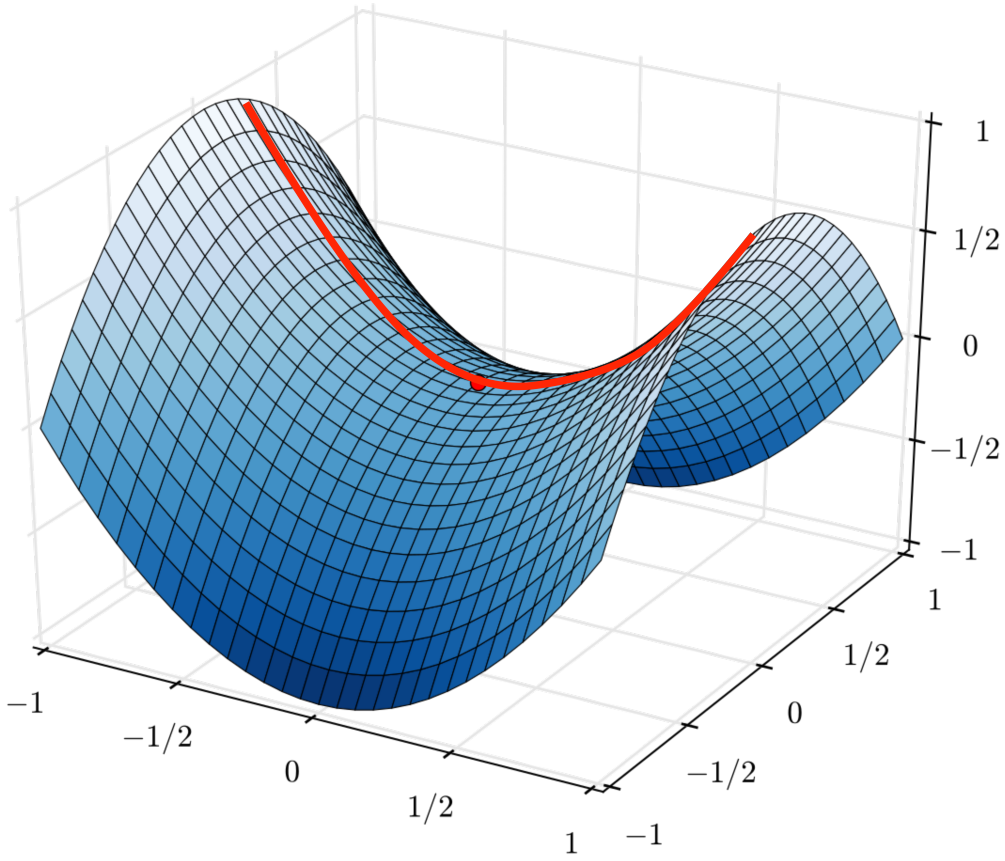
$$\max_{\boldsymbol{y}\in\mathbb{R}^n}\left\{D(\boldsymbol{y})\stackrel{\text{def}}{=}-g^*(-\frac{A^\top\boldsymbol{y}}{n})-\frac{1}{n}\sum_{i=1}^{n}\phi_i^*(y_i)\right\}$$

$\Rightarrow$ Lagrangian (convex-concave saddle point problem):

$$\max_{\boldsymbol{y}\in\mathbb{R}^n}\min_{\boldsymbol{x}\in\mathbb{R}^d}\left\{\mathcal{L}(\boldsymbol{x},\boldsymbol{y})\stackrel{\text{def}}{=}g(\boldsymbol{x})+\frac{1}{n}\boldsymbol{y}^\top A\boldsymbol{x}-\frac{1}{n}\sum_{i=1}^{n}\phi_i^*(y_i)\right\}$$

Illustration





Building up

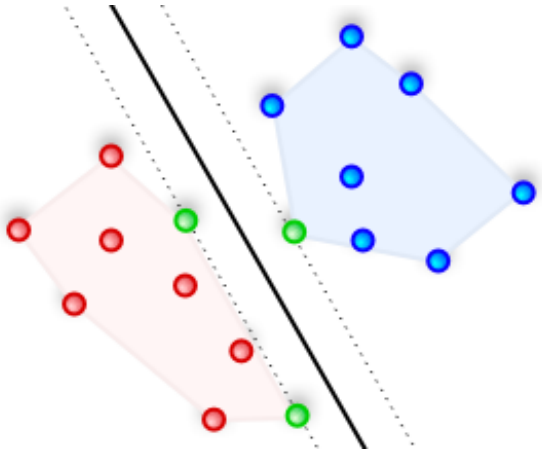
Why this convex-concave formulations?

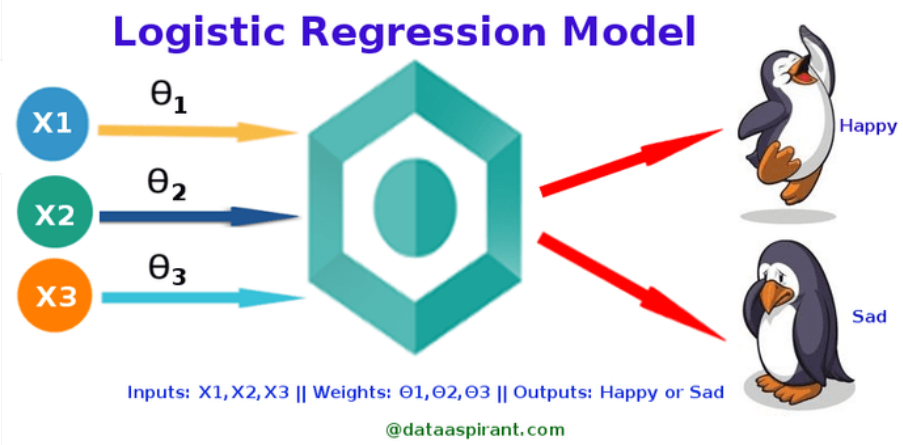
$$\min_{\boldsymbol{x}\in\mathbb{R}^d}\max_{\boldsymbol{y}\in\mathbb{R}^n}\left\{L(\boldsymbol{x},\boldsymbol{y})=f(\boldsymbol{x})+\boldsymbol{y}^\top A\boldsymbol{x}-g(\boldsymbol{y})\right\}$$

1. many machine learning applications

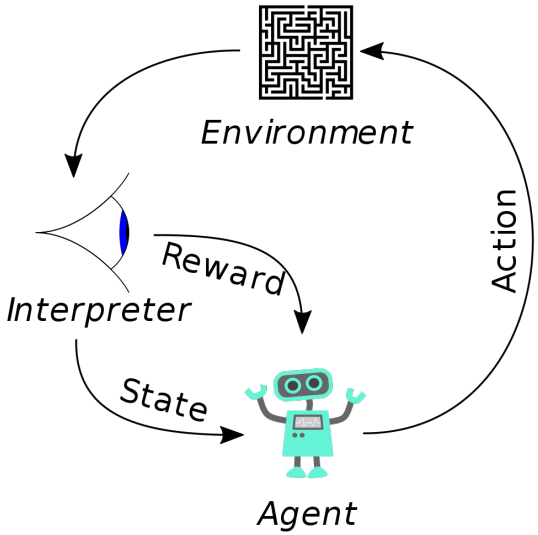
Machine Learning Applications

Empirical Risk Minimization



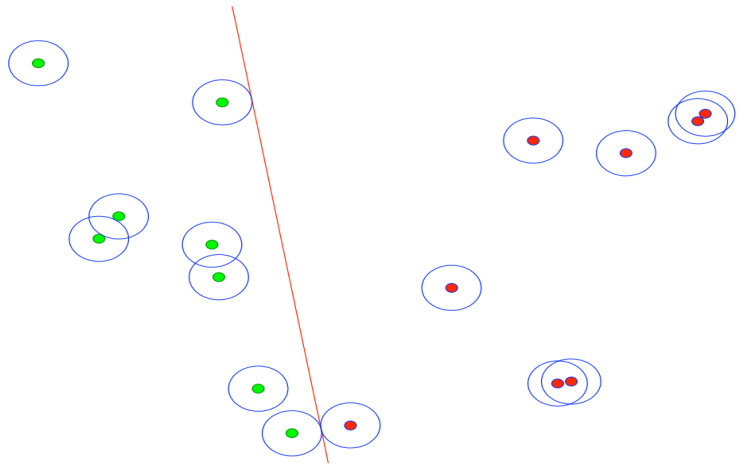


Reinforcement Learning



(Du et al. 2017)

Robust Optimization



(Ben-Tal et al., 2009)

Building up

Why this convex-concave formulations?

$$\min_{\boldsymbol{x}\in\mathbb{R}^d}\max_{\boldsymbol{y}\in\mathbb{R}^n}\left\{L(\boldsymbol{x},\boldsymbol{y})=f(\boldsymbol{x})+\boldsymbol{y}^\top A\boldsymbol{x}-g(\boldsymbol{y})\right\}$$

1. many machine learning applications
2. more suitable for greedy coordinate methods

A theoretical vignette

Q

What are some advantages of using greedy coordinate methods?

- Given a function $f(\boldsymbol{x})$, $\boldsymbol{x} \in \mathbb{R}^d$ that is μ-strongly convex, and L-smooth,
  1) CD with random selection/permutation: $\mathcal{O}(\frac{dL}{\mu} \log \frac{1}{\epsilon})$ iterations to achieve ϵ error.
  2) CD with greedy selection: $\mathcal{O}(\frac{L}{\mu_1} \log \frac{1}{\epsilon})$ iterations, if loss function is μ_1 ℓ_1-strongly convex (Nutini et al. 2015)

A theoretical vignette

Q

What are some advantages of using greedy coordinate methods?

- Given a function $f(\boldsymbol{x})$, $\boldsymbol{x} \in \mathbb{R}^d$ that is μ-strongly convex, and L-smooth,
  1) CD with random selection/permutation: $\mathcal{O}(\frac{dL}{\mu} \log \frac{1}{\epsilon})$ iterations to achieve ϵ error.
  2) CD with greedy selection: $\mathcal{O}(\frac{L}{\mu_1} \log \frac{1}{\epsilon})$ iterations, if loss function is μ_1 ℓ_1-strongly convex (Nutini et al. 2015)
  1. μ_1 could be as worse as μ/d
  2. A special case, when iterates and optimal values are s-sparse, it only requires $\mathcal{O}(\frac{sL}{\mu} \log \frac{1}{\epsilon})$ iterations to achieve ϵ error

A theoretical vignette

Q

What are some advantages of using greedy coordinate methods?

- Given a function $f(\boldsymbol{x})$, $\boldsymbol{x} \in \mathbb{R}^d$ that is μ-strongly convex, and L-smooth,
  1) CD with random selection/permutation: $\mathcal{O}(\frac{dL}{\mu} \log \frac{1}{\epsilon})$ iterations to achieve ϵ error.
  2) CD with greedy selection: $\mathcal{O}(\frac{L}{\mu_1} \log \frac{1}{\epsilon})$ iterations, if loss function is μ_1 ℓ_1-strongly convex (Nutini et al. 2015)
  1. μ_1 could be as worse as μ/d
  2. A special case, when iterates and optimal values are s-sparse, it only requires $\mathcal{O}(\frac{sL}{\mu} \log \frac{1}{\epsilon})$ iterations to achieve ϵ error
  3. Many applications have such primal sparsity:
     - Large-scale multiclass/multilabel
     - Low-degree polynomial data mapping
     - N-gram feature mapping
     - Random feature kernel machines

Problem with GCD on Primal

- Selection of greedy coordinate is expensive!!
- Naive implementation per iteration: $\mathcal{O}(nd)$
- Some heuristics NN (Dhillon et al. 2015), MIPS (Karimireddy et al. 2018) try to do the selection with time sublinear to d
- The only exception: gradient could be maintained: i.e. gradient is linear in x, $g(\boldsymbol{x}) = A\boldsymbol{x} + \mathbf{b}$. Or $P(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top A\boldsymbol{x} + \mathbf{b}^\top \boldsymbol{x}$
- since when $\boldsymbol{x}^+ \leftarrow \boldsymbol{x} + \delta_i \boldsymbol{e}_i$, $g^+ \leftarrow g + \delta_i \boldsymbol{a}_i$. Time complexity $\mathcal{O}(d)$

Primal-Dual formulation to the Rescue!

$$\min_{\boldsymbol{x}\in\mathbb{R}^d}\max_{\boldsymbol{y}\in\mathbb{R}^n}\left\{L(\boldsymbol{x},\boldsymbol{y})=f(\boldsymbol{x})+y^\top A\boldsymbol{x}-g(\boldsymbol{y})\right\}$$

Maintain $\boldsymbol{w}=A\boldsymbol{x}$ and $\boldsymbol{z}=A^\top\boldsymbol{y}$.

For each iteration,

Operation	cost
Compute full gradient $\partial_{\boldsymbol{y}}L=A\boldsymbol{x}-g'(\boldsymbol{y})$	$\mathcal{O}(nd)$
Choose a greedy coordinate j for $\boldsymbol{y}$	$\mathcal{O}(n)$
$\boldsymbol{y}^+\leftarrow\boldsymbol{y}+\delta_{\boldsymbol{y}}\boldsymbol{e}_j$	$\mathcal{O}(1)$
$\boldsymbol{z}^+\leftarrow\boldsymbol{z}+\delta_{\boldsymbol{y}}A^\top\boldsymbol{e}_j$	$\mathcal{O}(d)$
vice versa for $\boldsymbol{x}$ and $\boldsymbol{w}$	$\mathcal{O}(n+d)$

Primal-Dual formulation to the Rescue!

$$\min_{\boldsymbol{x}\in\mathbb{R}^d}\max_{\boldsymbol{y}\in\mathbb{R}^n}\left\{L(\boldsymbol{x},\boldsymbol{y})=f(\boldsymbol{x})+y^\top A\boldsymbol{x}-g(\boldsymbol{y})\right\}$$

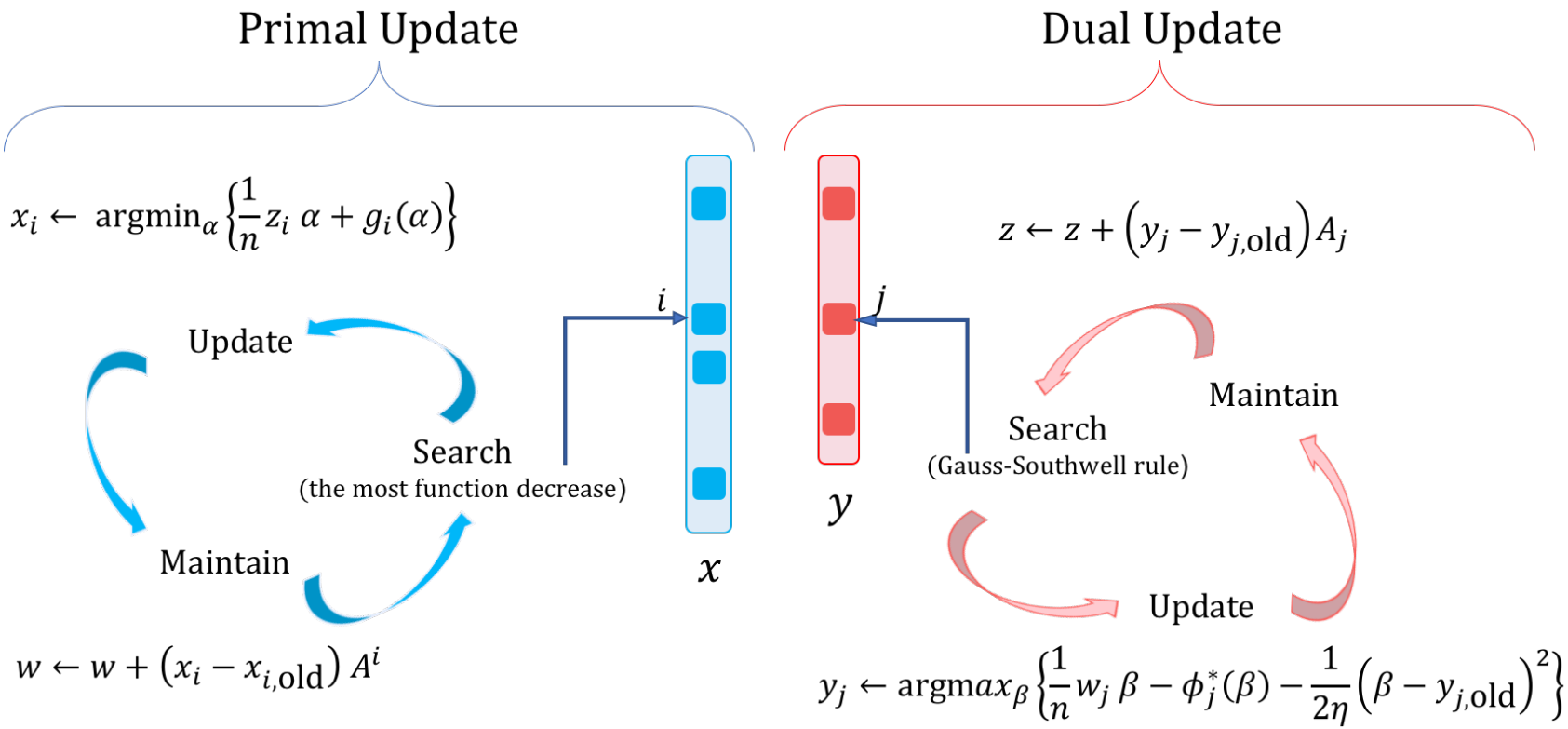
Maintain $\boldsymbol{w}=A\boldsymbol{x}$ and $\boldsymbol{z}=A^\top\boldsymbol{y}$.

For each iteration,

Operation	cost
Compute full gradient $\partial_{\boldsymbol{y}}L=\boldsymbol{w}-g'(\boldsymbol{y})$	$\mathcal{O}(n)$
Choose a greedy coordinate j for $\boldsymbol{y}$	$\mathcal{O}(n)$
$\boldsymbol{y}^+\leftarrow\boldsymbol{y}+\delta_{\boldsymbol{y}}\boldsymbol{e}_j$	$\mathcal{O}(1)$
$\boldsymbol{z}^+\leftarrow\boldsymbol{z}+\delta_{\boldsymbol{y}}A^\top\boldsymbol{e}_j$	$\mathcal{O}(d)$
vice versa for $\boldsymbol{x}$ and $\boldsymbol{w}$	$\mathcal{O}(n+d)$

Our method: Doubly Greedy Primal-Dual Coordinate methods

- Recall we want to achieve $\max_{\boldsymbol{y}\in\mathbb{R}^n}\min_{\boldsymbol{x}\in\mathbb{R}^d}\left\{\mathcal{L}(\boldsymbol{x},\boldsymbol{y})\overset{\text{def}}{=}g(\boldsymbol{x})+\frac{1}{n}\boldsymbol{y}^\top A\boldsymbol{x}-\frac{1}{n}\sum_{i=1}^n\phi_i^*(y_i)\right\}$
- Start from $\boldsymbol{x}\leftarrow 0,\boldsymbol{y}\leftarrow 0,\boldsymbol{w}(\equiv A\boldsymbol{x})\leftarrow 0\in\mathbb{R}^n,\boldsymbol{z}(\equiv A^\top\boldsymbol{y})\leftarrow 0\in\mathbb{R}^d$.
- $\mathcal{O}(d+n)$ operations per iteration



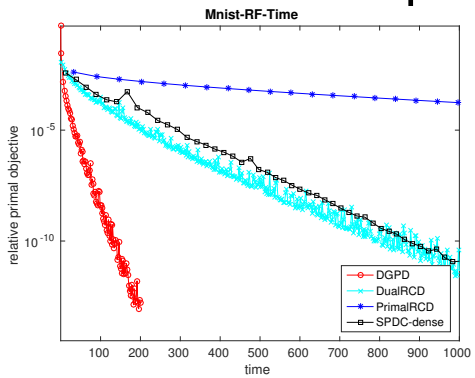
Time complexity comparisons

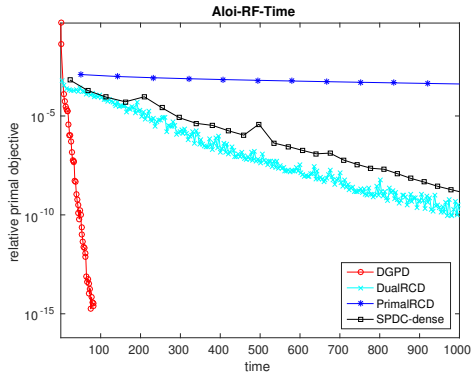
method	per Iteration	# Iteration
GD	$\mathcal{O}(nd)$	$\mathcal{O}(\kappa \log^{-1} \epsilon)$
RCD	$\mathcal{O}(n)$	$\mathcal{O}(d\kappa \log^{-1} \epsilon)$
SGD	$\mathcal{O}(d)$	$\mathcal{O}(\kappa\epsilon^{-1})$
MISO	$\mathcal{O}(d)$	$\mathcal{O}((n+\kappa) \log^{-1} \epsilon)$
SVRG		
SAG		$\mathcal{O}((n+\sqrt{n\kappa}) \log^{-1} \epsilon)$
SAGA		if accelerated
SDCA	$\mathcal{O}(d)$	$\mathcal{O}((n+\kappa) \log^{-1} \epsilon)$
SPDC	$\mathcal{O}(d)$	$\mathcal{O}((n+\sqrt{n\kappa}) \log^{-1} \epsilon)$
DGPDC	$\mathcal{O}(d+n)$	$\mathcal{O}(\frac{s}{n}(n+\kappa) \log^{-1} \epsilon)$

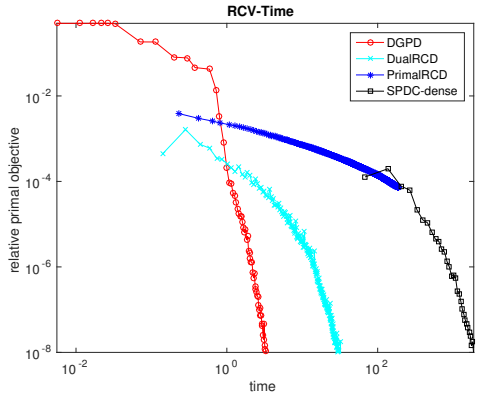
- DGPDC (Lei et al. 2017)

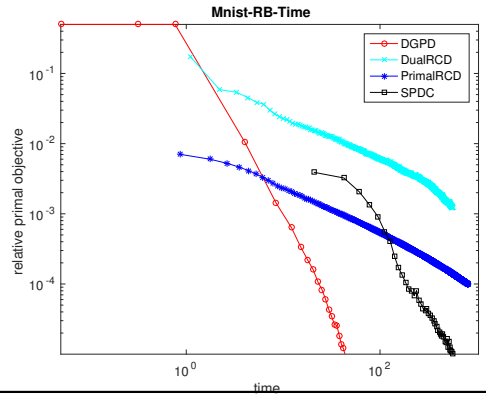
Experiments

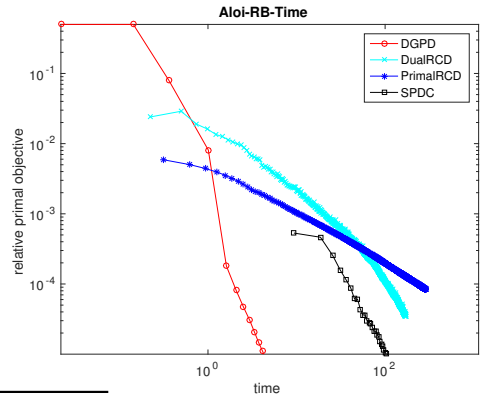
- Compared methods:
  - Stochastic Dual Coordinate Ascent (SDCA, or DualRCD)
  - Primal Randomize Coordinate Descent (PrimalRCD)
  - Stochastic Primal-Dual Coordinate method (SPDC)
- Performance comparisons:[1]

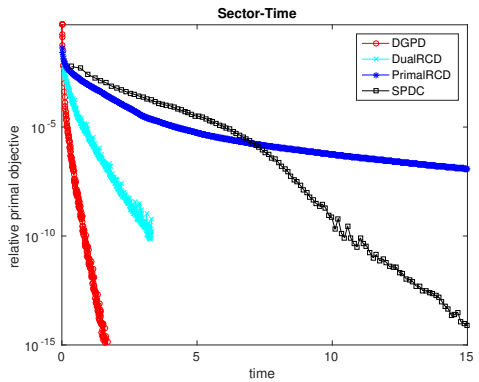












[1] Relative objective versus time for $\lambda = 0.1, \mu = 0.01$. All datasets are downloaded from LIBSVM `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`

Other trial on the Primal-Dual formulation

$$\min_{\boldsymbol{x}\in\mathbb{R}^d}\max_{\boldsymbol{y}\in\mathbb{R}^n}\left\{L(\boldsymbol{x},\boldsymbol{y})=f(\boldsymbol{x})+\boldsymbol{y}^\top A\boldsymbol{x}-g(\boldsymbol{y})\right\}$$

method	per Iteration	# Iteration
SDCA	$\mathcal{O}(d)$	$\mathcal{O}((n+\kappa)\log^{-1}\epsilon)$
SPDC	$\mathcal{O}(d)$	$\mathcal{O}((n+\sqrt{n\kappa})\log^{-1}\epsilon)$
DGPDC (ours)	$\mathcal{O}(d+n)$	$\mathcal{O}((s+\frac{s}{n}\kappa)\log^{-1}\epsilon)$
DSPDC	$\mathcal{O}(d)$	$\mathcal{O}((n+k\sqrt{n\kappa_1})\log\frac{1}{\epsilon})$

- DSPDC (Yu et al. 2015), assuming data is factorized with rank k. $\kappa_1 \overset{\text{def}}{=} \frac{\|\boldsymbol{a}_i\|_\infty^2}{\lambda\mu} \in [\frac{\kappa}{d},\kappa]$

Q&A

Thank you!

Datasets

Table: Data statistics and number of non-zero primal & dual variables from DGPD (w/ $\lambda = 0.1$, $\mu = 0.01$).

Data set	#features	#nz/sample	#train samples	#test samples	#classes	#nz-primal	#nz-dual
Mnist-RF	10,000	10,000	58,000	2,000	10	1,730	2,000
Aloi-RF	10,000	10,000	90,000	8,000	1,000	891	1,428
Mnist-RB	1,572,556	1,000	58,000	2,000	10	1,733	1,208
Aloi-RB	636,910	200	90,000	8,000	1,000	1,032	782
RCV1-Regions	47,236	68.38	199,328	23,149	225	1,123	1,447
Sector	55,197	162.94	7,793	961	105	610	655