Exploiting Strong Convexity from Data with Primal-Dual First-Order Algorithms

Jialei Wang[1] **Lin Xiao**[2]

Abstract

We consider empirical risk minimization of linear predictors with convex loss functions. Such problems can be reformulated as convex-concave saddle point problems and solved by primal-dual first-order algorithms. However, primal-dual algorithms often require explicit strongly convex regularization in order to obtain fast linear convergence, and the required dual proximal mapping may not admit closed-form or efficient solution. In this paper, we develop both batch and randomized primal-dual algorithms that can exploit strong convexity from data adaptively and are capable of achieving linear convergence even without regularization. We also present dual-free variants of adaptive primal-dual algorithms that do not need the dual proximal mapping, which are especially suitable for logistic regression.

1. Introduction

We consider the problem of regularized empirical risk minimization (ERM) of linear predictors. Let $a_1, \ldots, a_n \in \mathbb{R}^d$ be the feature vectors of n data samples, $\phi_i : \mathbb{R} \to \mathbb{R}$ be a convex loss function associated with the linear prediction $a_i^T x$, for $i = 1, \ldots, n$, and $g : \mathbb{R}^d \to \mathbb{R}$ be a convex regularization function for the predictor $x \in \mathbb{R}^d$. ERM amounts to solving the following convex optimization problem:

$$\min_{x\in\mathbb{R}^d} \quad \left\{ P(x) \stackrel{\text{def}}{=} \tfrac{1}{n}\textstyle\sum_{i=1}^n \phi_i(a_i^T x) + g(x) \right\}. \qquad (1)$$

This formulation covers many well-known classification and regression problems. For example, logistic regression is obtained by setting $\phi_i(z) = \log(1 + \exp(-b_i z))$ where $b_i \in \{\pm 1\}$. For linear regression problems, the loss function is $\phi_i(z) = (1/2)(z - b_i)^2$, and we get ridge regression with $g(x) = (\lambda/2)\|x\|_2^2$ and the elastic net with $g(x) = \lambda_1\|x\|_1 + (\lambda_2/2)\|x\|_2^2$.

Let $A = [a_1, \ldots, a_n]^T$ be the n by d data matrix. Throughout this paper, we make the following assumptions:

Assumption 1. *The functions ϕ_i, g and matrix A satisfy:*

- *Each ϕ_i is δ-strongly convex and $1/\gamma$-smooth where $\gamma > 0$ and $\delta \geq 0$, and $\gamma\delta \leq 1$;*
- *g is λ-strongly convex where $\lambda \geq 0$;*
- *$\lambda + \delta\mu^2 > 0$, where $\mu = \sqrt{\lambda_{\min}(A^T A)}$.*

The strong convexity and smoothness mentioned above are with respect to the standard Euclidean norm, denoted as $\|x\| = \sqrt{x^T x}$. (See, e.g., Nesterov (2004, Sections 2.1.1 and 2.1.3) for the exact definitions.) We allow $\delta = 0$, which simply means ϕ_i is convex. Let $R = \max_i\{\|a_i\|\}$ and assuming $\lambda > 0$, then $R^2/(\gamma\lambda)$ is a popular definition of condition number for analyzing complexities of different algorithms. The last condition above means that the primal objective function $P(x)$ is strongly convex, even if $\lambda = 0$.

There have been extensive research activities in recent years on developing efficiently algorithms for solving problem (1). A broad class of randomized algorithms that exploit the finite sum structure in the ERM problem have emerged as very competitive both in terms of theoretical complexity and practical performance. They can be put into three categories: primal, dual, and primal-dual.

Primal randomized algorithms work with the ERM problem (1) directly. They are modern versions of randomized incremental gradient methods (e.g., Bertsekas, 2012; Nedic & Bertsekas, 2001) equipped with variance reduction techniques. Each iteration of such algorithms only process one data point a_i with complexity $O(d)$. They includes SAG (Roux et al., 2012), SAGA (Defazio et al., 2014), and SVRG (Johnson & Zhang, 2013; Xiao & Zhang, 2014), which all achieve the iteration complexity $O\left((n + R^2/(\gamma\lambda))\log(1/\epsilon)\right)$ to find an ϵ-optimal solution. In fact, they are capable of exploiting the strong convexity from data, meaning that the condition number $R^2/(\gamma\lambda)$ in the complexity can be replaced by the more favorable one $R^2/(\gamma(\lambda+\delta\mu^2/n))$. This improvement can be achieved without explicit knowledge of μ from data.

[1] Department of Computer Science, The University of Chicago, Chicago, Illinois 60637, USA. [2] Microsoft Research, Redmond, Washington 98052, USA. Correspondence to: Jialei Wang <jialei@uchicago.edu>, Lin Xiao <lin.xiao@microsoft.com>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017. Copyright 2017 by the author(s).

Dual algorithms solve Fenchel dual of (1) by maximizing

$$D(y) \overset{\text{def}}{=} \tfrac{1}{n}\textstyle\sum_{i=1}^n -\phi_i^*(y_i) - g^*\left(-\tfrac{1}{n}\sum_{i=1}^n y_i a_i\right) \qquad (2)$$

using randomized coordinate ascent algorithms. (Here ϕ_i^* and g^* denotes the conjugate functions of ϕ_i and g.) They include SDCA (Shalev-Shwartz & Zhang, 2013), Nesterov (2012) and Richtárik & Takáč (2014). They have the same complexity $O\left((n + R^2/(\gamma\lambda))\log(1/\epsilon)\right)$, but cannot exploit strong convexity, if any (when $\delta\mu^2 > 0$), from data.

Primal-dual algorithms solve the convex-concave saddle point problem $\min_x \max_y \mathcal{L}(x, y)$ where

$$\mathcal{L}(x, y) \overset{\text{def}}{=} \tfrac{1}{n}\textstyle\sum_{i=1}^n \big(y_i\langle a_i, x\rangle - \phi_i^*(y_i)\big) + g(x). \qquad (3)$$

In particular, SPDC (Zhang & Xiao, 2015) achieves an accelerated linear convergence rate with iteration complexity $O\left((n + \sqrt{n}R/\sqrt{\gamma\lambda})\log(1/\epsilon)\right)$, which is better than the aforementioned non-accelerated complexity when $R^2/(\gamma\lambda) > n$. Lan & Zhou (2015) developed dual-free variants of accelerated primal-dual algorithms, but without considering the linear predictor structure in ERM. Balamurugan & Bach (2016) extended SVRG and SAGA to solving saddle point problems.

Accelerated primal and dual randomized algorithms have also been developed. Nesterov (2012), Fercoq & Richtárik (2015) and Lin et al. (2015b) developed accelerated coordinate gradient algorithms, which can be applied to solve the dual problem (2). Allen-Zhu (2016) developed an accelerated variant of SVRG. Acceleration can also be obtained using the Catalyst framework (Lin et al., 2015a). They all achieve the same $O\left((n + \sqrt{n}R/\sqrt{\gamma\lambda})\log(1/\epsilon)\right)$ complexity. A common feature of accelerated algorithms is that they require good estimate of the strong convexity parameter. This makes hard for them to exploit strong convexity from data because the minimum singular value μ of the data matrix A is very hard to estimate in general.

In this paper, we show that primal-dual algorithms are capable of exploiting strong convexity from data if the algorithm parameters (such as step sizes) are set appropriately. While these optimal setting depends on the knowledge of the convexity parameter μ from the data, we develop adaptive variants of primal-dual algorithms that can tune the parameter automatically. Such adaptive schemes rely critically on the capability of evaluating the primal-dual optimality gaps by primal-dual algorithms.

A major disadvantage of primal-dual algorithms is that the required dual proximal mapping may not admit closed-form or efficient solution. We follow the approach of Lan & Zhou (2015) to derive dual-free variants of the primal-dual algorithms customized for ERM problems with the linear predictor structure, and show that they can also exploit strong convexity from data with correct choices of parameters or using an adaptation scheme.

Algorithm 1 Batch Primal-Dual (BPD) Algorithm

```
input: parameters $\tau$, $\sigma$, $\theta$, initial point $(\tilde{x}^{(0)} = x^{(0)}, y^{(0)})$
  for $t = 0, 1, 2, \ldots$ do
    $y^{(t+1)} = \mathrm{prox}_{\sigma f^*}\left(y^{(t)} + \sigma A\tilde{x}^{(t)}\right)$
    $x^{(t+1)} = \mathrm{prox}_{\tau g}\left(x^{(t)} - \tau A^T y^{(t+1)}\right)$
    $\tilde{x}^{(t+1)} = x^{(t+1)} + \theta(x^{(t+1)} - x^{(t)})$
  end for
```

2. Batch primal-dual algorithms

We first study batch primal-dual algorithms, by considering a "batch" version of the ERM problem (1),

$$\min_{x\in\mathbb{R}^d} \left\{P(x) \overset{\text{def}}{=} f(Ax) + g(x)\right\}. \qquad (4)$$

where $A \in \mathbb{R}^{n\times d}$. We make the following assumptions:

Assumption 2. *The functions f, g and matrix A satisfy:*

- *f is δ-strongly convex and $1/\gamma$-smooth where $\gamma > 0$ and $\delta \geq 0$, and $\gamma\delta \leq 1$;*
- *g is λ-strongly convex where $\lambda \geq 0$;*
- *$\lambda + \delta\mu^2 > 0$, where $\mu = \sqrt{\lambda_{\min}(A^T A)}$.*

Using conjugate functions, we can derive the dual of (4) as

$$\max_{y\in\mathbb{R}^n} \left\{D(y) \overset{\text{def}}{=} -f^*(y) - g^*(-A^T y)\right\}, \qquad (5)$$

and the convex-concave saddle point formulation is

$$\min_{x\in\mathbb{R}^d} \max_{y\in\mathbb{R}^n} \left\{\mathcal{L}(x, y) \overset{\text{def}}{=} g(x) + y^T A x - f^*(y)\right\}. \qquad (6)$$

We consider the primal-dual first-order algorithm proposed by Chambolle & Pock (2011; 2016) for solving the saddle point problem (6), given in Algorithm 1, where $\mathrm{prox}_\psi(\cdot)$, for any convex function $\psi : \mathbb{R}^n \cup \{\infty\}$, is defined as

$$\mathrm{prox}_\psi(\beta) = \arg\min_{\alpha\in\mathbb{R}^n} \left(\psi(\alpha) + (1/2)\|\alpha - \beta\|^2\right).$$

Assuming that f is smooth and g is strongly convex, Chambolle & Pock (2011; 2016) showed that Algorithm 1 achieves accelerated linear convergence rate if $\lambda > 0$. However, they did not consider the case where additional or the sole source of strong convexity comes from $f(Ax)$. In the following theorem, we show how to set the parameters τ, σ and θ to exploit both sources of strong convexity to achieve fast linear convergence.

Theorem 1. *Suppose Assumption 2 holds and $(x^\star, y^\star)$ is the unique saddle point of $\mathcal{L}$ defined in (6). Let $L = \|A\| = \sqrt{\lambda_{\max}(A^T A)}$. If we set the parameters in Algorithm 1 as*

$$\sigma = \tfrac{1}{L}\sqrt{\tfrac{\lambda+\delta\mu^2}{\gamma}}, \qquad \tau = \tfrac{1}{L}\sqrt{\tfrac{\gamma}{\lambda+\delta\mu^2}}, \qquad (7)$$

and $\theta = \max\{\theta_x, \theta_y\}$ where

$$\theta_x = \left(1 - \tfrac{\delta}{(\delta+2\sigma)}\tfrac{\mu^2}{L^2}\right)\tfrac{1}{1+\tau\lambda}, \qquad \theta_y = \tfrac{1}{1+\sigma\gamma/2}, \qquad (8)$$

then we have

$$\left(\tfrac{1}{2\tau}+\tfrac{\lambda}{2}\right)\|x^{(t)}-x^\star\|^2+\tfrac{\gamma}{4}\|y^{(t)}-y^\star\|^2 \le \theta^t C,$$
$$\mathcal{L}(x^{(t)},y^\star)-\mathcal{L}(x^\star,y^{(t)}) \le \theta^t C,$$

where $C = \left(\frac{1}{2\tau}+\frac{\lambda}{2}\right)\|x^{(0)}-x^\star\|^2+\left(\frac{1}{2\sigma}+\frac{\gamma}{4}\right)\|y^{(0)}-y^\star\|^2$.

The proof of Theorem 1 is given in Appendices B and C. Here we give a detailed analysis of the convergence rate. Substituting σ and τ in (7) into the expressions for θ_y and θ_x in (8), and assuming $\gamma(\lambda+\delta\mu^2) \ll L^2$, we have

$$\theta_x \approx 1-\frac{\gamma\delta\mu^2}{L^2}\left(2\frac{\sqrt{\gamma(\lambda+\delta\mu^2)}}{L}+\gamma\delta\right)^{-1}-\frac{\lambda}{L}\sqrt{\frac{\gamma}{\lambda+\delta\mu^2}},$$
$$\theta_y = \frac{1}{1+\sqrt{\gamma(\lambda+\delta\mu^2)}/(2L)} \approx 1-\frac{\sqrt{\gamma(\lambda+\delta\mu^2)}}{2L}.$$

Since the overall condition number of the problem is $\frac{L^2}{\gamma(\lambda+\delta\mu^2)}$, it is clear that θ_y is an accelerated convergence rate. Next we examine θ_x in two special cases.

The case of $\delta\mu^2 = 0$ but $\lambda > 0$. In this case, we have $\tau = \frac{1}{L}\sqrt{\frac{\gamma}{\lambda}}$ and $\sigma = \frac{1}{L}\sqrt{\frac{\lambda}{\gamma}}$, and thus

$$\theta_x = \frac{1}{1+\sqrt{\gamma\lambda}/L} \approx 1-\frac{\sqrt{\gamma\lambda}}{L}, \quad \theta_y = \frac{1}{1+\sqrt{\gamma\lambda}/(2L)} \approx 1-\frac{\sqrt{\gamma\lambda}}{2L}.$$

Therefore we have $\theta = \max\{\theta_x, \theta_y\} \approx 1-\frac{\sqrt{\lambda\gamma}}{2L}$. This indeed is an accelerated convergence rate, recovering the result of Chambolle & Pock (2011; 2016).

The case of $\lambda = 0$ but $\delta\mu^2 > 0$. In this case, we have $\tau = \frac{1}{L\mu}\sqrt{\frac{\gamma}{\delta}}$ and $\sigma = \frac{\mu}{L}\sqrt{\frac{\delta}{\gamma}}$, and

$$\theta_x = 1-\frac{\gamma\delta\mu^2}{L^2}\cdot\frac{1}{2\sqrt{\gamma\delta}\mu/L+\gamma\delta}, \quad \theta_y \approx 1-\frac{\sqrt{\gamma\delta}\mu}{2L}.$$

Notice that $\frac{1}{\gamma\delta}\frac{L^2}{\mu^2}$ is the condition number of $f(Ax)$. Next we assume $\mu \ll L$ and examine how θ_x varies with $\gamma\delta$.

- If $\gamma\delta \approx \frac{\mu^2}{L^2}$, meaning f is badly conditioned, then
$$\theta_x \approx 1-\frac{\gamma\delta\mu^2}{L^2}\cdot\frac{1}{3\sqrt{\gamma\delta}\mu/L} = 1-\frac{\sqrt{\gamma\delta}\mu}{3L}.$$
Because the overall condition number is $\frac{1}{\gamma\delta}\frac{L^2}{\mu^2}$, this is an accelerated linear rate, and so is $\theta = \max\{\theta_x, \theta_y\}$.
- If $\gamma\delta \approx \frac{\mu}{L}$, meaning f is mildly conditioned, then
$$\theta_x \approx 1-\frac{\mu^3}{L^3}\frac{1}{2(\mu/L)^{3/2}+\mu/L} \approx 1-\frac{\mu^2}{L^2}.$$
This represents a half-accelerated rate, because the overall condition number is $\frac{1}{\gamma\delta}\frac{L^2}{\mu^2} \approx \frac{L^3}{\mu^3}$.
- If $\gamma\delta = 1$, i.e., f is a simple quadratic function, then
$$\theta_x \approx 1-\frac{\mu^2}{L^2}\frac{1}{2\mu/L+1} \approx 1-\frac{\mu^2}{L^2}.$$
This rate does not have acceleration, because the overall condition number is $\frac{1}{\gamma\delta}\frac{L^2}{\mu^2} \approx \frac{L^2}{\mu^2}$.

Algorithm 2 Adaptive Batch Primal-Dual (Ada-BPD)

```
input: problem constants $\lambda$, $\gamma$, $\delta$, $L$ and $\hat{\mu} > 0$, initial
       point $(x^{(0)}, y^{(0)})$, and adaptation period $T$.
  Compute $\sigma$, $\tau$, and $\theta$ as in (7) and (8) using $\mu = \hat{\mu}$
  for $t = 0, 1, 2, \ldots$ do
    $y^{(t+1)} = \mathrm{prox}_{\sigma f^*}\left(y^{(t)} + \sigma A\tilde{x}^{(t)}\right)$
    $x^{(t+1)} = \mathrm{prox}_{\tau g}\left(x^{(t)} - \tau A^T y^{(t+1)}\right)$
    $\tilde{x}^{(t+1)} = x^{(t+1)} + \theta(x^{(t+1)} - x^{(t)})$
    if $\mathrm{mod}(t+1, T) == 0$ then
      $(\sigma, \tau, \theta) = $ BPD-Adapt $\left(\{P^{(s)}, D^{(s)}\}_{s=t-T}^{t+1}\right)$
    end if
  end for
```

Algorithm 3 BPD-Adapt (simple heuristic)

```
input: previous estimate $\hat{\mu}$, adaption period $T$, primal and
       dual objective values $\{P^{(s)}, D^{(s)}\}_{s=t-T}^{t}$
  if $P^{(t)} - D^{(t)} < \theta^T(P^{(t-T)} - D^{(t-T)})$ then
    $\hat{\mu} := \sqrt{2}\hat{\mu}$
  else
    $\hat{\mu} := \hat{\mu}/\sqrt{2}$
  end if
  Compute $\sigma$, $\tau$, and $\theta$ as in (7) and (8) using $\mu = \hat{\mu}$
output: new parameters $(\sigma, \tau, \theta)$
```

In summary, the extent of acceleration in the dominating factor θ_x (which determines θ) depends on the relative size of $\gamma\delta$ and μ^2/L^2, i.e., the relative conditioning between the function f and the matrix A. In general, we have full acceleration if $\gamma\delta \le \mu^2/L^2$. The theory predicts that the acceleration degrades as the function f gets better conditioned. However, in our numerical experiments, we often observe acceleration even if $\gamma\delta$ gets closer to 1.

As explained in Chambolle & Pock (2011), Algorithm 1 is equivalent to a preconditioned ADMM. Deng & Yin (2016) characterized various conditions for ADMM to obtain linear convergence, but did not derive the convergence rate for the case we consider in this paper.

2.1. Adaptive batch primal-dual algorithms

In practice, it is often very hard to obtain a good estimate of the problem-dependent constants, especially $\mu = \sqrt{\lambda_{\min}(A^T A)}$, in order to apply the algorithmic parameters specified in Theorem 1. Here we explore heuristics that can enable adaptive tuning of such parameters, which often lead to much improved performance in practice.

A key observation is that the convergence rate of the BPD algorithm changes monotonically with the overall convexity parameter $\lambda + \delta\mu^2$, regardless of the extent of acceleration. In other words, the larger $\lambda + \delta\mu^2$ is, the faster the convergence. Therefore, if we can monitor the progress of

Algorithm 4 BPD-Adapt (robust heuristic)

```
input: previous rate estimate $\rho > 0$, $\Delta = \delta\hat{\mu}^2$, period $T$,
       constants $\underline{c} < 1$ and $\overline{c} > 1$, and $\{P^{(s)}, D^{(s)}\}_{s=t-T}^{t}$
  Compute new rate estimate $\hat{\rho} = \frac{P^{(t)}-D^{(t)}}{P^{(t-T)}-D^{(t-T)}}$
  if $\hat{\rho} \le \underline{c}\,\rho$ then
    $\Delta := 2\Delta, \qquad \rho := \hat{\rho}$
  else if $\hat{\rho} \ge \overline{c}\,\rho$ then
    $\Delta := \Delta/2, \qquad \rho := \hat{\rho}$
  else
    $\Delta := \Delta$
  end if
  $\sigma = \frac{1}{L}\sqrt{\frac{\lambda+\Delta}{\gamma}}, \quad \tau = \frac{1}{L}\sqrt{\frac{\gamma}{\lambda+\Delta}}$
  Compute $\theta$ using (8) or set $\theta = 1$
output: new parameters $(\sigma, \tau, \theta)$
```

the convergence and compare it with the predicted convergence rate in Theorem 1, then we can adjust the estimated parameters to exploit strong convexity from data. More specifically, if the observed convergence rate is slower than the predicted rate, then we should reduce the estimate of μ; otherwise we should increase μ for faster convergence.

We formalize the above reasoning in Algorithm 2 (called Ada-BPD). This algorithm maintains an estimate $\hat{\mu}$ of the true constant μ, and adjust it every T iterations. We use $P^{(t)}$ and $D^{(t)}$ to represent the primal and dual objective values at $P(x^{(t)})$ and $D(y^{(t)})$, respectively. We give two implementations of the tuning procedure BPD-Adapt: Algorithm 3 is a simple heuristic for tuning the estimate $\hat{\mu}$, where the increasing and decreasing factor $\sqrt{2}$ can be changed to other values larger than 1. Algorithm 4 is a more robust heuristic. It does not rely on the specific convergence rate θ established in Theorem 1. Instead, it simply compares the current estimate of objective reduction rate $\hat{\rho}$ with the previous estimate ρ. It also specifies a non-tuning range of changes in ρ, specified by the interval $[\underline{c}, \overline{c}]$.

The capability of accessing both the primal and dual objective values allows primal-dual algorithms to have good estimate of the convergence rate, which enables effective tuning heuristics. Automatic tuning of primal-dual algorithms have also been studied by, e.g., Malitsky & Pock (2016) and Goldstein et al. (2013), but with different goals.

Algorithm 5 Adaptive SPDC (Ada-SPDC)

```
input: parameters $\sigma$, $\tau$, $\theta > 0$, initial point $(x^{(0)}, y^{(0)})$,
       and adaptation period $T$.
  Set $\tilde{x}^{(0)} = x^{(0)}$
  for $t = 0, 1, 2, \ldots$ do
    pick $k \in \{1, \ldots, n\}$ uniformly at random
    for $i \in \{1, \ldots, n\}$ do
      if $i == k$ then
        $y_k^{(t+1)} = \mathrm{prox}_{\sigma\phi_k^*}\left(y_k^{(t)} + \sigma a_k^T \tilde{x}^{(t)}\right)$
      else
        $y_i^{(t+1)} = y_i^{(t)}$
      end if
    end for
    $x^{(t+1)} = \mathrm{prox}_{\tau g}\left(x^{(t)} - \tau\big(u^{(t)} + (y_k^{(t+1)} - y_k^{(t)})a_k\big)\right)$
    $u^{(t+1)} = u^{(t)} + \frac{1}{n}(y_k^{(t+1)} - y_k^{(t)})a_k$
    $\tilde{x}^{(t+1)} = x^{(t+1)} + \theta(x^{(t+1)} - x^{(t)})$
    if $\mathrm{mod}(t+1, T \cdot n) = 0$ then
      $(\tau, \sigma, \theta) = \text{SPDC-Adapt}\big(\{P^{(t-sn)}, D^{(t-sn)}\}_{s=0}^{T}\big)$
    end if
  end for
```

3. Randomized primal-dual algorithm

In this section, we come back to the ERM problem and consider its saddle-point formulation in (3). Due to its finite sum structure in the dual variables y_i, we can develope randomized algorithms to exploit strong convexity from data. In particular, we extend the stochastic primal-dual coordinate (SPDC) algorithm by Zhang & Xiao (2015). SPDC is a special case of the Ada-SPDC algorithm in Algorithm 5, by setting the adaption period $T = \infty$ (no adaption). The following theorem is proved in Appendix E.

Theorem 2. *Suppose Assumption 1 holds. Let $(x^\star, y^\star)$ be the saddle point of the function $\mathcal{L}$ defined in (3), and $R = \max\{\|a_1\|, \ldots, \|a_n\|\}$. If we set $T = \infty$ in Algorithm 5 (no adaption) and let*

$$\tau = \frac{1}{4R}\sqrt{\frac{\gamma}{n\lambda+\delta\mu^2}}, \quad \sigma = \frac{1}{4R}\sqrt{\frac{n\lambda+\delta\mu^2}{\gamma}}, \tag{9}$$

and $\theta = \max\{\theta_x, \theta_y\}$ where

$$\theta_x = \left(1-\frac{\tau\sigma\delta\mu^2}{2n(\sigma+4\delta)}\right)\frac{1}{1+\tau\lambda}, \quad \theta_y = \frac{1+((n-1)/n)\sigma\gamma/2}{1+\sigma\gamma/2}, \tag{10}$$

then we have

$$\begin{aligned}\left(\tfrac{1}{2\tau}+\tfrac{\lambda}{2}\right)\mathbb{E}\big[\|x^{(t)}-x^\star\|^2\big] + \tfrac{\gamma}{4}\mathbb{E}\big[\|y^{(t)}-y^\star\|^2\big] &\le \theta^t C,\\ \mathbb{E}\left[\mathcal{L}(x^{(t)},y^\star)-\mathcal{L}(x^\star,y^{(t)})\right] &\le \theta^t C,\end{aligned}$$

where $C = \left(\frac{1}{2\tau}+\frac{\lambda}{2}\right)\|x^{(0)}-x^\star\|^2 + \left(\frac{1}{2\sigma}+\frac{\gamma}{4}\right)\|y^{(0)}-y^\star\|^2$. The expectation $\mathbb{E}[\cdot]$ is taken with respect to the history of random indices drawn at each iteration.

Below we give a detailed discussion on the expected convergence rate established in Theorem 2.

The cases of $\sigma\mu^2 = 0$ but $\lambda > 0$. In this case we have $\tau = \frac{1}{4R}\sqrt{\frac{\gamma}{n\lambda}}$ and $\sigma = \frac{1}{4R}\sqrt{\frac{n\lambda}{\gamma}}$, and

$$\theta_x = \frac{1}{1+\tau\lambda} = 1 - \frac{1}{1+4R\sqrt{n/(\lambda\gamma)}},$$

$$\theta_y = \frac{1+((n-1)/n)\sigma\gamma/2}{1+\sigma\gamma/2} = 1 - \frac{1}{n+8R\sqrt{n/(\lambda\gamma)}}.$$

Hence $\theta = \theta_y$. These recover the parameters and convergence rate of the standard SPDC (Zhang & Xiao, 2015).

The cases of $\sigma\mu^2 > 0$ but $\lambda = 0$. In this case we have $\tau = \frac{1}{4R\mu}\sqrt{\frac{\gamma}{\delta}}$ and $\sigma = \frac{\mu}{4R}\sqrt{\frac{\delta}{\gamma}}$, and

$$\theta_x = 1 - \frac{\tau\sigma\delta\mu^2}{2n(\sigma+4\delta)} = 1 - \frac{\gamma\delta\mu^2}{32nR^2} \cdot \frac{1}{\sqrt{\gamma\delta}\mu/(4R)+4\gamma\delta}.$$

$$\theta_y = 1 - \frac{1}{n+8nR/(\mu\sqrt{\gamma\delta})} \approx 1 - \frac{\sqrt{\gamma\delta}\mu}{8nR}\left(1+\frac{\sqrt{\gamma\delta}\mu}{8R}\right)^{-1}.$$

Since the objective is R^2/γ-smooth and $\delta\mu^2/n$-strongly convex, θ_y is an accelerated rate if $\frac{\sqrt{\gamma\delta}\mu}{8R} \ll 1$ (otherwise $\theta_y \approx 1 - \frac{1}{n}$). For θ_x, we consider different situations:

- If $\mu \geq R$, then we have $\theta_x \approx 1 - \frac{\sqrt{\gamma\delta}\mu}{nR}$, which is an accelerated rate. So is $\theta = \max\{\theta_x, \theta_y\}$.
- If $\mu < R$ and $\gamma\delta \approx \frac{\mu^2}{R^2}$, then $\theta_x \approx 1 - \frac{\sqrt{\gamma\delta}\mu}{nR}$, which represents an accelerated rate. The iteration complexity of SPDC is $\widetilde{O}(\frac{nR}{\mu\sqrt{\gamma\delta}})$, which is better than that of SVRG in this case, which is $\widetilde{O}(\frac{nR^2}{\gamma\delta\mu^2})$.
- If $\mu < R$ and $\gamma\delta \approx \frac{\mu}{R}$, then we get $\theta_x \approx 1 - \frac{\mu^2}{nR^2}$. This is a half-accelerated rate, because in this case SVRG requires $\widetilde{O}(\frac{nR^3}{\mu^3})$ iterations, versus $\widetilde{O}(\frac{nR^2}{\mu^2})$ for SPDC.
- If $\mu < R$ and $\gamma\delta \approx 1$, meaning the ϕ_i's are well conditioned, then we get $\theta_x \approx 1 - \frac{\gamma\delta\mu^2}{nR^2} \approx 1 - \frac{\mu^2}{nR^2}$, which is a non-accelerated rate. The corresponding iteration complexity is the same as SVRG.

3.1. Parameter adaptation for SPDC

The SPDC-Adapt procedure called in Algorithm 5 follows the same logics as the batch adaption schemes in Algorithms 3 and 4, and we omit the details here. One thing we emphasize here is that the adaptation period T is in terms of epochs, or number of passes over the data. In addition, we only compute the primal and dual objective values after each pass or every few passes, because computing them exactly usually need to take a full pass of the data.

Unlike the batch case where the duality gap decreases monotonically, the duality gap for randomized algorithms can fluctuate wildly. So instead of using only the two end values $P^{(t-Tn)} - D^{(t-Tn)}$ and $P^{(t)} - D^{(t)}$, we can use more points to estimate the convergence rate through a linear regression. Suppose the primal-dual objective values for the last $T+1$ passes are $(P(0), D(0)), (P(1), D(1)), \ldots, (P(T), D(T))$, and we need to estimate ρ (rate per pass) such that

$$P(t) - D(t) \approx \rho^t\big(P(0) - D(0)\big), \quad t = 1, \ldots, T.$$

We can turn it into a linear regression problem after taking logarithm and obtain the estimate $\hat{\rho}$ through

$$\log(\hat{\rho}) = \frac{1}{1^2+2^2+\cdots+T^2}\sum_{t=1}^{T} t \log\frac{P(t)-D(t)}{P(0)-D(0)}.$$

Algorithm 6 Dual-Free BPD Algorithm

```
input: parameters $\sigma, \tau, \theta > 0$, initial point $(x^{(0)}, y^{(0)})$
  Set $\tilde{x}^{(0)} = x^{(0)}$ and $v^{(0)} = (f^*)'(y^{(0)})$
  for $t = 0, 1, 2, \ldots$ do
    $v^{(t+1)} = \frac{v^{(t)}+\sigma A\tilde{x}^{(t)}}{1+\sigma}, \quad y^{(t+1)} = f'(v^{(t+1)})$
    $x^{(t+1)} = \mathrm{prox}_{\tau g}\left(x^{(t)} - \tau A^T y^{(t+1)}\right)$
    $\tilde{x}^{(t+1)} = x^{(t+1)} + \theta(x^{(t+1)} - x^{(t)})$
  end for
```

4. Dual-free Primal-dual algorithms

Compared with primal algorithms, one major disadvantage of primal-dual algorithms is the requirement of computing the proximal mapping of the dual function f^* or ϕ_i^*, which may not admit closed-formed solution or efficient computation. This is especially the case for logistic regression, one of the most popular loss functions used in classification.

Lan & Zhou (2015) developed "dual-free" variants of primal-dual algorithms that avoid computing the dual proximal mapping. Their main technique is to replace the Euclidean distance in the dual proximal mapping with a Bregman divergence defined over the dual loss function itself. We show how to apply this approach to solve the structured ERM problems considered in this paper. They can also exploit strong convexity from data if the algorithmic parameters are set appropriately or adapted automatically.

4.1. Dual-free BPD algorithm

First, we consider the batch setting. We replace the dual proximal mapping (computing $y^{(t+1)}$) in Algorithm 1 with

$$y^{(t+1)} = \arg\min_y \left\{f^*(y) - y^T A\tilde{x}^{(t)} + \tfrac{1}{\sigma}\mathcal{D}(y, y^{(t)})\right\}, \quad (11)$$

where $\mathcal{D}$ is the Bregman divergence of a strictly convex kernel function h, defined as

$$\mathcal{D}_h(y, y^{(t)}) = h(y) - h(y^{(t)}) - \langle\nabla h(y^{(t)}), y - y^{(t)}\rangle.$$

Algorithm 1 is obtained in the Euclidean setting with $h(y) = \frac{1}{2}\|y\|^2$ and $\mathcal{D}(y, y^{(t)}) = \frac{1}{2}\|y - y^{(t)}\|^2$. Here we use f^* as the kernel function, and show that it allows us to compute $y^{(t+1)}$ in (11) very efficiently. The following lemma explains the details (Cf. Lan & Zhou, 2015, Lemma 1).

Lemma 1. *Let the kernel $h \equiv f^*$ in the Bregman divergence $\mathcal{D}$. If we construct a sequence of vectors $\{v^{(t)}\}$ such that $v^{(0)} = (f^*)'(y^{(0)})$ and for all $t \geq 0$,*

$$v^{(t+1)} = \frac{v^{(t)}+\sigma A\tilde{x}^{(t)}}{1+\sigma}, \quad (12)$$

then the solution to problem (11) is $y^{(t+1)} = f'(v^{(t+1)})$.

Proof. Suppose $v^{(t)} = (f^*)'(y^{(t)})$ (true for $t = 0$), then

$$\mathcal{D}(y, y^{(t)}) = f^*(y) - f^*(y^{(t)}) - {v^{(t)}}^T(y - y^{(t)}).$$

The solution to (11) can be written as

$$\begin{aligned} y^{(t+1)} &= \arg\min_y \Big\{ f^*(y) - y^T A\tilde{x}^{(t)} + \tfrac{1}{\sigma}\big(f^*(y) - {v^{(t)}}^T y\big) \Big\} \\ &= \arg\min_y \Big\{ \big(1+\tfrac{1}{\sigma}\big) f^*(y) - \big(A\tilde{x}^{(t)} + \tfrac{1}{\sigma} v^{(t)}\big)^T y \Big\} \\ &= \arg\max_y \Big\{ \Big(\tfrac{v^{(t)}+\sigma A\tilde{x}^{(t)}}{1+\sigma}\Big)^T y - f^*(y) \Big\} \\ &= \arg\max_y \Big\{ {v^{(t+1)}}^T y - f^*(y) \Big\} = f'(v^{(t+1)}), \end{aligned}$$

where in the last equality we used the property of conjugate function when f is strongly convex and smooth. Moreover,

$$v^{(t+1)} = (f')^{-1}(y^{(t+1)}) = (f^*)'(y^{(t+1)}),$$

which completes the proof. □

According to Lemma 1, we only need to provide initial points such that $v^{(0)} = (f^*)'(y^{(0)})$ is easy to compute. We do not need to compute $(f^*)'(y^{(t)})$ directly for any $t > 0$, because it is can be updated as $v^{(t)}$ in (12). Consequently, we can update $y^{(t)}$ in the BPD algorithm using the gradient $f'(v^{(t)})$, without the need of dual proximal mapping. The resulting dual-free algorithm is given in Algorithm 6.

Theorem 3. *Suppose Assumption 2 holds and let $(x^\star, y^\star)$ be the unique saddle point of $\mathcal{L}$ defined in (6). If we set the parameters in Algorithm 6 as*

$$\tau = \tfrac{1}{L}\sqrt{\tfrac{\gamma}{\lambda+\delta\mu^2}}, \quad \sigma = \tfrac{1}{L}\sqrt{\gamma(\lambda+\delta\mu^2)}, \tag{13}$$

and $\theta = \max\{\theta_x, \theta_y\}$ where

$$\theta_x = \Big(1 - \tfrac{\tau\sigma\delta\mu^2}{(4+2\sigma)}\Big)\tfrac{1}{1+\tau\lambda}, \quad \theta_y = \tfrac{1}{1+\sigma/2}, \tag{14}$$

then we have

$$\begin{aligned} \big(\tfrac{1}{2\tau} + \tfrac{\lambda}{2}\big)\|x^{(t)} - x^\star\|^2 + \tfrac{1}{2}\mathcal{D}(y^\star, y^{(t)}) &\le \theta^t C, \\ \mathcal{L}(x^{(t)}, y^\star) - \mathcal{L}(x^\star, y^{(t)}) &\le \theta^t C, \end{aligned}$$

where $C = \big(\frac{1}{2\tau}+\frac{\lambda}{2}\big)\|x^{(0)} - x^\star\|^2 + \big(\frac{1}{\sigma}+\frac{1}{2}\big)\mathcal{D}(y^\star, y^{(0)})$.

Theorem 3 is proved in Appendices B and D. Assuming $\gamma(\lambda+\delta\mu^2) \ll L^2$, we have

$$\theta_x \approx 1 - \tfrac{\gamma\delta\mu^2}{16L^2} - \tfrac{\lambda}{2L}\sqrt{\tfrac{\gamma}{\lambda+\delta\mu^2}}, \quad \theta_y \approx 1 - \tfrac{\sqrt{\gamma(\lambda+\delta\mu^2)}}{4L}.$$

Again, we gain insights by consider the special cases:

- If $\delta\mu^2 = 0$ and $\lambda > 0$, then $\theta_y \approx 1 - \frac{\sqrt{\gamma\lambda}}{4L}$ and $\theta_x \approx 1 - \frac{\sqrt{\gamma\lambda}}{2L}$. So $\theta = \max\{\theta_x, \theta_y\}$ is an accelerated rate.
- If $\delta\mu^2 > 0$ and $\lambda = 0$, then $\theta_y \approx 1 - \frac{\sqrt{\gamma\delta\mu^2}}{4L}$ and $\theta_x \approx 1 - \frac{\gamma\delta\mu^2}{16L^2}$. Thus $\theta = \max\{\theta_x, \theta_y\} \approx 1 - \frac{\gamma\delta\mu^2}{16L^2}$ is not accelerated. This conclusion does not depends on the relative sizes of $\gamma\delta$ and μ^2/L^2, and it is the major difference from the Euclidean case in Section 2.

Algorithm 7 Adaptive Dual-Free SPDC (ADF-SPDC)

```
input: parameters $\sigma, \tau, \theta > 0$, initial point $(x^{(0)}, y^{(0)})$,
       and adaptation period $T$.
  Set $\tilde{x}^{(0)} = x^{(0)}$ and $v_i^{(0)} = (\phi_i^*)'(y_i^{(0)})$ for $i = 1, \ldots, n$
  for $t = 0, 1, 2, \ldots$ do
    pick $k \in \{1, \ldots, n\}$ uniformly at random
    for $i \in \{1, \ldots, n\}$ do
      if $i == k$ then
        $v_k^{(t+1)} = \frac{v_k^{(t)} + \sigma a_k^T \tilde{x}^{(t)}}{1+\sigma}, \quad y_k^{(t+1)} = \phi_k'(v_k^{(t+1)})$
      else
        $v_i^{(t+1)} = v_i^{(t)}, \quad y_i^{(t+1)} = y_i^{(t)}$
      end if
    end for
    $x^{(t+1)} = \mathrm{prox}_{\tau g}\Big(x^{(t)} - \tau\big(u^{(t)} + (y_k^{(t+1)} - y_k^{(t)})a_k\big)\Big)$
    $u^{(t+1)} = u^{(t)} + \frac{1}{n}(y_k^{(t+1)} - y_k^{(t)})a_k$
    $\tilde{x}^{(t+1)} = x^{(t+1)} + \theta(x^{(t+1)} - x^{(t)})$
    if $\mathrm{mod}(t+1, T \cdot n) = 0$ then
      $(\tau, \sigma, \theta) = \text{SPDC-Adapt}\big(\{P^{(t-sn)}, D^{(t-sn)}\}_{s=0}^{T}\big)$
    end if
  end for
```

If both $\delta\mu^2 > 0$ and $\lambda > 0$, then the extent of acceleration depends on their relative size. If λ is on the same order as $\delta\mu^2$ or larger, then accelerated rate is obtained. If λ is much smaller than $\delta\mu^2$, then the theory predicts no acceleration.

4.2. Dual-free SPDC algorithm

Following the same approach, we can derive an Adaptive Dual-Free SPDC algorithm, given in Algorithm 7. On related work, Shalev-Shwartz & Zhang (2016) and (Shalev-Shwartz, 2016) introduced dual-free SDCA.

The following theorem characterizes the choice of algorithmic parameters that can exploit strong convexity from data to achieve linear convergence (proof given in Appendix F).

Theorem 4. *Suppose Assumption 1 holds. Let $(x^\star, y^\star)$ be the saddle point of $\mathcal{L}$ in (3) and $R = \max\{\|a_1\|, \ldots, \|a_n\|\}$. If we set $T = \infty$ in Algorithm 7 (non adaption) and let*

$$\sigma = \tfrac{1}{4R}\sqrt{\gamma(n\lambda+\delta\mu^2)}, \quad \tau = \tfrac{1}{4R}\sqrt{\tfrac{\gamma}{n\lambda+\delta\mu^2}}, \tag{15}$$

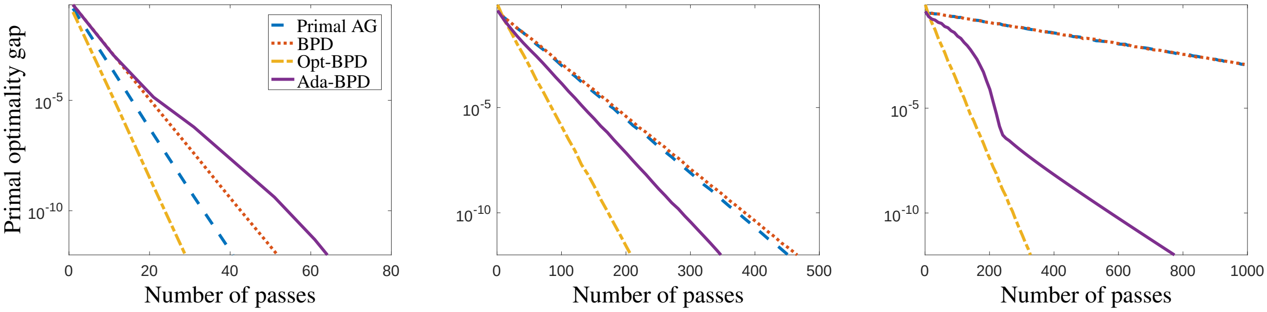
and $\theta = \max\{\theta_x, \theta_y\}$ where

$$\theta_x = \Big(1 - \tfrac{\tau\sigma\delta\mu^2}{n(4+2\sigma)}\Big)\tfrac{1}{1+\tau\lambda}, \quad \theta_y = \tfrac{1+((n-1)/n)\sigma/2}{1+\sigma/2}, \tag{16}$$

then we have

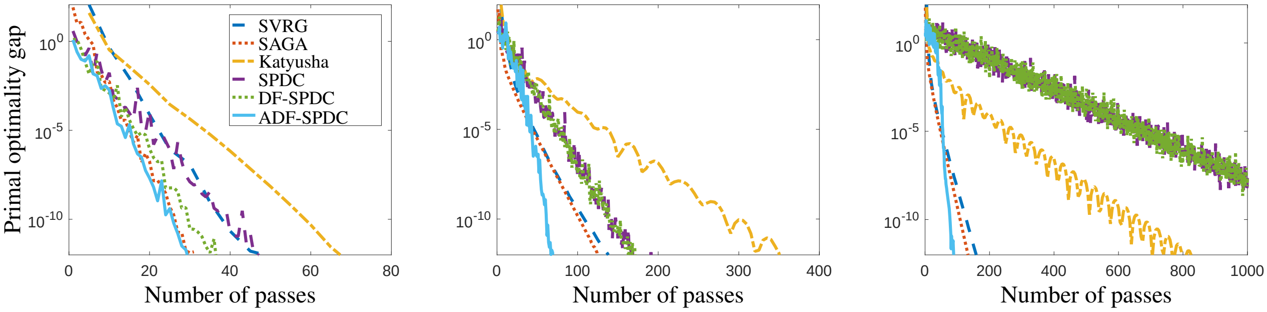
$$\begin{aligned} \big(\tfrac{1}{2\tau} + \tfrac{\lambda}{2}\big)\mathbb{E}\big[\|x^{(t)} - x^\star\|^2\big] + \tfrac{\gamma}{4}\mathbb{E}\big[\mathcal{D}(y^\star, y^{(t)})\big] &\le \theta^t C, \\ \mathbb{E}\big[\mathcal{L}(x^{(t)}, y^\star) - \mathcal{L}(x^\star, y^{(t)})\big] &\le \theta^t C, \end{aligned}$$

where $C = \big(\frac{1}{2\tau}+\frac{\lambda}{2}\big)\|x^{(0)} - x^\star\|^2 + \big(\frac{1}{\sigma}+\frac{1}{2}\big)\mathcal{D}(y^\star, y^{(0)})$.



synthetic1, $\lambda = 1/n$ synthetic1, $\lambda = 10^{-2}/n$ synthetic1, $\lambda = 10^{-4}/n$

Figure 1. Comparison of batch primal-dual algorithms for a ridge regression problem with $n = 5000$ and $d = 3000$.



cpuact, $\lambda = 1/n$ cpuact, $\lambda = 10^{-2}/n$ cpuact, $\lambda = 10^{-4}/n$

Figure 2. Comparison of randomized algorithms for ridge regression problems.

Now we discuss the results of Theorem 4 in further details.

The cases of $\sigma\mu^2 = 0$ but $\lambda > 0$. In this case we have $\tau = \frac{1}{4R}\sqrt{\frac{\gamma}{n\lambda}}$ and $\sigma = \frac{1}{4R}\sqrt{n\gamma\lambda}$, and

$$\theta_x = 1 - \frac{1}{1+4R\sqrt{n/(\lambda\gamma)}}, \qquad \theta_y = 1 - \frac{1}{n+8R\sqrt{n/(\lambda\gamma)}}.$$

The rate is the same as for SPDC in Zhang & Xiao (2015).

The cases of $\sigma\mu^2 > 0$ but $\lambda = 0$. In this case we have $\tau = \frac{1}{4R\mu}\sqrt{\frac{\gamma}{\delta}}$ and $\sigma = \frac{\mu}{4R}\sqrt{\delta\gamma}$, thus

$$\theta_x = 1 - \frac{\tau\sigma\delta\mu^2}{2n(\sigma+4)} = 1 - \frac{\gamma\delta\mu^2}{32nR^2} \cdot \frac{1}{\sqrt{\gamma\delta}\mu/(4R)+4},$$

$$\theta_y = \frac{1+((n-1)/n)\sigma/2}{1+\sigma/2} = 1 - \frac{1}{n+8nR/(\mu\sqrt{\gamma\delta})}.$$

We note that the primal function now is R^2/γ-smooth and $\delta\mu^2/n$-strongly convex. We discuss the following cases:

- If $\sqrt{\gamma\delta}\mu > R$, then we have $\theta_x \approx 1 - \frac{\sqrt{\gamma\delta}\mu}{8nR}$ and $\theta_y \approx 1 - \frac{1}{n}$. Therefore $\theta = \max\{\theta_x, \theta_y\} \approx 1 - \frac{1}{n}$.
- Otherwise, we have $\theta_x \approx 1 - \frac{\gamma\delta\mu^2}{64nR^2}$ and θ_y is of the same order. This is not an accelerated rate, and we have the same iteration complexity as SVRG.

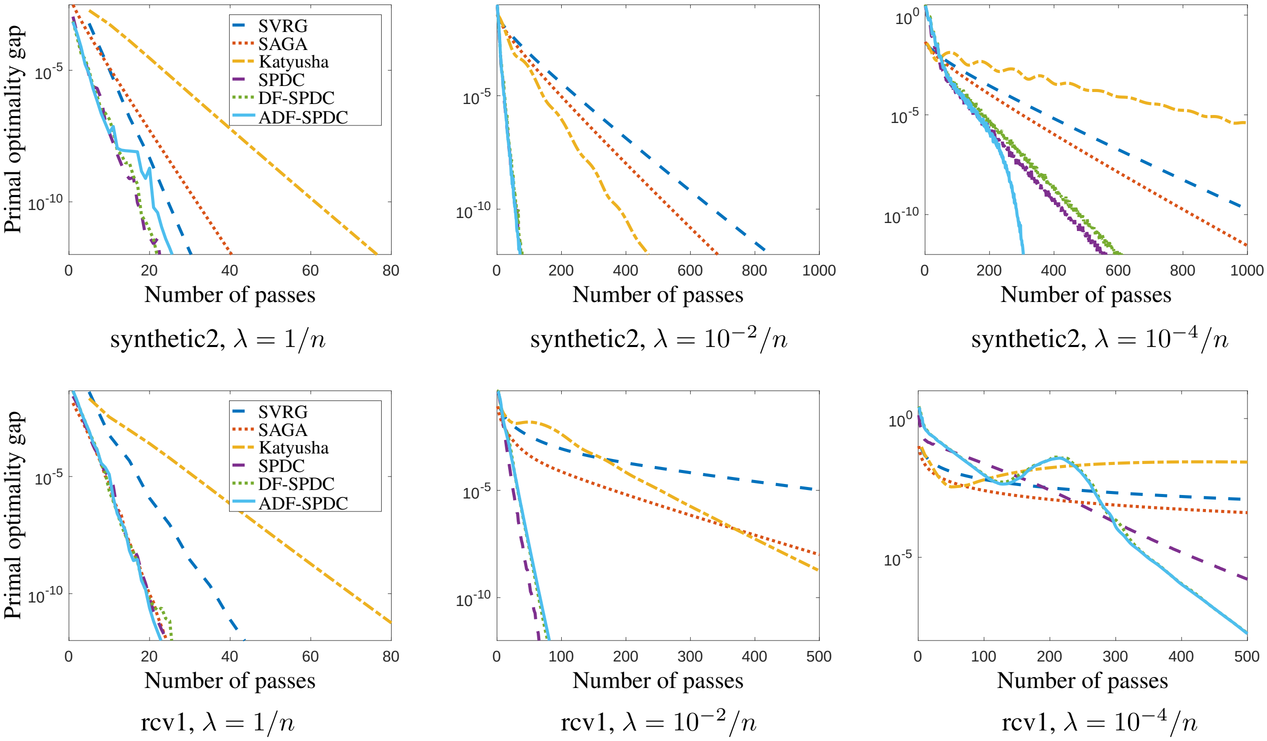
Finally, we give concrete examples of how to compute the initial points $y^{(0)}$ and $v^{(0)}$ such that $v_i^{(0)} = (\phi_i^*)'(y_i^{(0)})$.

- For squared loss, $\phi_i(\alpha) = \frac{1}{2}(\alpha - b_i)^2$ and $\phi_i^*(\beta) = \frac{1}{2}\beta^2 + b_i\beta$. So $v_i^{(0)} = (\phi_i^*)'(y_i^{(0)}) = y_i^{(0)} + b_i$.
- For logistic regression, we have $b_i \in \{1, -1\}$ and $\phi_i(\alpha) = \log(1 + e^{-b_i\alpha})$. The conjugate function is $\phi_i^*(\beta) = (-b_i\beta)\log(-b_i\beta) + (1 + b_i\beta)\log(1 + b_i\beta)$ if $b_i\beta \in [-1, 0]$ and $+\infty$ otherwise. We can choose $y_i^{(0)} = -\frac{1}{2}b_i$ and $v_i^{(0)} = 0$ such that $v_i^{(0)} = (\phi_i^*)'(y_i^{(0)})$.

For logistic regression, we have $\delta = 0$ over the full domain of ϕ_i. However, each ϕ_i is locally strongly convex in bounded domain (Bach, 2014): if $z \in [-B, B]$, then we know $\delta = \min_z \phi_i''(z) \geq \exp(-B)/4$. Therefore it is well suitable for an adaptation scheme similar to Algorithm 4 that do not require knowledge of either δ or μ.

5. Preliminary experiments

We present preliminary experiments to demonstrate the effectiveness of our proposed algorithms. First, we consider batch primal-dual algorithms for ridge regression over a synthetic dataset. The data matrix A has sizes $n = 5000$ and $d = 3000$, and its entries are sampled from multivariate normal distribution with mean zero and covariance matrix $\Sigma_{ij} = 2^{|i-j|/2}$. We normalize all datasets



synthetic2, $\lambda = 1/n$ synthetic2, $\lambda = 10^{-2}/n$ synthetic2, $\lambda = 10^{-4}/n$

rcv1, $\lambda = 1/n$ rcv1, $\lambda = 10^{-2}/n$ rcv1, $\lambda = 10^{-4}/n$

Figure 3. Comparison of randomized algorithms for logistic regression problems.

such that $a_i = a_i / (\max_j \|a_j\|)$, to ensure the maximum norm of the data points is 1. We use ℓ_2-regularization $g(x) = (\lambda/2)\|x\|^2$ with three choices of parameter λ: $1/n$, $10^{-2}/n$ and $10^{-4}/n$, which represent the strong, medium, and weak levels of regularization, respectively.

Figure 1 shows the performance of four different algorithms: the primal accelerated gradient (Primal AG) algorithm (Nesterov, 2004) using λ as strong convexity parameter, the BPD algorithm (Algorithm 1) that uses the same λ and $\mu^2\delta = 0$, the optimal BPD algorithm (Opt-BPD) that uses $\mu^2\delta = \frac{\lambda_{\min}(A^TA)}{n} \approx \frac{0.022}{n}$ computed from data, and the Ada-BPD algorithm (Algorithm 2) with the robust adaptation heuristic (Algorithm 4) with $T = 10$, $\underline{c} = 0.95$ and $\overline{c} = 1.5$. As expected, the performance of Primal-AG is very similar to that of BPD, and Opt-BPD has the fastest convergence. The Ada-BPD algorithm can partially exploit strong convexity from data without knowledge of μ.

Next we compare DF-SPDC (Algorithm 5 without adaption) and ADF-SPDC (Algorithm 7 with adaption) against several state-of-the-art randomized algorithms for ERM: SVRG (Johnson & Zhang, 2013), SAGA (Defazio et al., 2014) Katyusha (Allen-Zhu, 2016) and the standard SPDC method (Zhang & Xiao, 2015). For SVRG and Katyusha (an accelerated variant of SVRG), we choose the variance reduction period as $m = 2n$. The step sizes of all algorithms are set as their original paper suggested. For Ada-SPDC and ADF-SPDC, we use the robust adaptation scheme with $T = 10$, $\underline{c} = 0.95$ and $\overline{c} = 1.5$.

We first compare these randomized algorithms for ridge regression over the `cpuact` data from the LibSVM website (https://www.csie.ntu.edu.tw/~cjlin/libsvm/). The results are shown in Figure 2. With relatively strong regularization $\lambda = 1/n$, all methods perform similarly as predicted by theory. When λ becomes smaller, the non-accelerated algorithms (SVRG and SAGA) automatically exploit strong convexity from data, so they become faster than the non-adaptive accelerated methods (Katyusha, SPDC and DF-SPDC). The adaptive accelerated method, ADF-SPDC, has the fastest convergence. This indicates that our theoretical results, which predict no acceleration in this case, may be further improved.

Finally we compare these algorithms for logistic regression on the `rcv1` dataset (from LibSVM website) and another synthetic dataset with $n = 5000$ and $d = 500$, generated similarly as before but with covariance matrix $\Sigma_{ij} = 2^{|i-j|/100}$. For the standard SPDC, we compute the coordinate-wise dual proximal mapping using a few steps of scalar Newton's method to high precision. The dual-free SPDC algorithms only use gradients of the logistic function. The results are presented in Figure 3. For both datasets, the strong convexity from data is very weak, and the accelerated algorithms performs better.

References

Allen-Zhu, Zeyuan. Katyusha: Accelerated variance reduction for faster sgd. *ArXiv e-print 1603.05953*, 2016.

Bach, Francis. Adaptivity of averaged stochastic gradient descent to local strong convexity for logistic regression. *Journal of Machine Learning Research*, 15(1):595–627, 2014.

Balamurugan, Palaniappan and Bach, Francis. Stochastic variance reduction methods for saddle-point problems. In *Advances in Neural Information Processing Systems (NIPS) 29*, pp. 1416–1424, 2016.

Bertsekas, Dimitri P. Incremental gradient, subgradient, and proximal methods for convex optimization: A survey. In Sra, Suvrit, Nowozin, Sebastian, and Wright, Stephen J. (eds.), *Optimization for Machine Learning*, chapter 4, pp. 85–120. MIT Press, 2012.

Chambolle, Antonin and Pock, Thomas. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

Chambolle, Antonin and Pock, Thomas. On the ergodic convergence rates of a first-order primal–dual algorithm. *Mathematical Programming, Series A*, 159:253–287, 2016.

Defazio, Aaron, Bach, Francis, and Lacoste-Julien, Simon. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pp. 1646–1654, 2014.

Deng, Wei and Yin, Wotao. On the global and linear convergence of the generalized alternating direction method of multipliers. *Journal of Scientific Computing*, 66(3): 889–916, 2016.

Fercoq, Oliver and Richtárik, Peter. Accelerated, parallel, and proximal coordinate descent. *SIAM Journal on Optimization*, 25(4):1997–2023, 2015.

Goldstein, Tom, Li, Min, Yuan, Xiaoming, Esser, Ernie, and Baraniuk, Richard. Adaptive primal-dual hybrid gradient methods for saddle-point problems. *arXiv preprint arXiv:1305.0546*, 2013.

Johnson, Rie and Zhang, Tong. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pp. 315–323, 2013.

Lan, Guanghui and Zhou, Yi. An optimal randomized incremental gradient method. *arXiv preprint arXiv:1507.02000*, 2015.

Lin, Hongzhou, Mairal, Julien, and Harchaoui, Zaid. A universal catalyst for first-order optimization. In *Advances in Neural Information Processing Systems*, pp. 3384–3392, 2015a.

Lin, Qihang, Lu, Zhaosong, and Xiao, Lin. An accelerated randomized proximal coordinate gradient method and its application to regularized empirical risk minimization. *SIAM Journal on Optimization*, 25(4):2244–2273, 2015b.

Malitsky, Yura and Pock, Thomas. A first-order primal-dual algorithm with linesearch. *arXiv preprint arXiv:1608.08883*, 2016.

Nedic, Angelia and Bertsekas, Dimitri P. Incremental subgradient methods for nondifferentiable optimization. *SIAM Journal on Optimization*, 12(1):109–138, 2001.

Nesterov, Yurii. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer, Boston, 2004.

Nesterov, Yurii. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Richtárik, Peter and Takáč, Martin. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 144(1-2):1–38, 2014.

Roux, Nicolas L, Schmidt, Mark, and Bach, Francis. A stochastic gradient method with an exponential convergence _rate for finite training sets. In *Advances in Neural Information Processing Systems*, pp. 2663–2671, 2012.

Shalev-Shwartz, Shai. SDCA without duality, regularization, and individual convexity. In *Proceedings of The 33rd International Conference on Machine Learning*, pp. 747–754, 2016.

Shalev-Shwartz, Shai and Zhang, Tong. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14(Feb): 567–599, 2013.

Shalev-Shwartz, Shai and Zhang, Tong. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. *Mathematical Programming*, 155(1-2): 105–145, 2016.

Xiao, Lin and Zhang, Tong. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

Zhang, Yuchen and Xiao, Lin. Stochastic primal-dual coordinate method for regularized empirical risk minimization. In *Proceedings of The 32nd International Conference on Machine Learning*, pp. 353–361, 2015.