Sub-First-Order Methods

Stephen J. Wright[1]

[2]Computer Sciences Department,
University of Wisconsin-Madison.

AMSI Winter School, July 2017

Stochastic Gradient Methods

Deal with (weakly or strongly) convex f.

- Allow f nonsmooth.
- Can't get function values $f(x)$ easily.
- At any feasible x, have access only to a cheap unbiased estimate of an element of the subgradient ∂f.

Common settings are:

$$f(x) = E_\xi F(x, \xi),$$

where ξ is a random vector with distribution P over a set Ξ. Special case:

$$f(x) = \frac{1}{m} \sum_{i=1}^{m} f_i(x),$$

where each f_i is convex and nonsmooth.
(We focus on this finite-sum formulation, but the ideas generalize.)

Applications

This setting is useful for machine learning formulations. Given data $x_i \in \mathbb{R}^n$ and labels $y_i = \pm 1$, $i = 1, 2, \dots, m$, find w that minimizes

$$\tau \psi(w) + \frac{1}{m} \sum_{i=1}^{m} \ell(w; x_i, y_i),$$

where ψ is a regularizer, $\tau > 0$ is a parameter, and ℓ is a loss. For linear classifiers/regressors, have the specific form $\ell(w^T x_i, y_i)$.

Example: SVM with hinge loss $\ell(w^T x_i, y_i) = \max(1 - y_i(w^T x_i), 0)$ and $\psi = \| \cdot \|_1$ or $\psi = \| \cdot \|_2^2$.

Example: Logistic regression: $\ell(w^T x_i, y_i) = \log(1 + \exp(y_i w^T x_i))$. In regularized version may have $\psi(w) = \|w\|_1$.

Example: Deep Learning (the killer app!).

Subgradients

Recall: For each x in domain of f, g is a *subgradient of f at x* if

$$f(z) \geq f(x) + g^T(z-x), \qquad \text{for all } z \in \operatorname{dom} f.$$

- Right-hand side is a *supporting hyperplane*.
- The set of subgradients is called the *subdifferential*, denoted by $\partial f(x)$.
- When f is differentiable at x, have $\partial f(x) = \{\nabla f(x)\}$.

We have strong convexity with modulus $m > 0$ if

$$f(z) \geq f(x) + g^T(z-x) + \frac{1}{2}m\|z-x\|^2, \quad \text{for all } x, z \in \operatorname{dom} f \text{ with } g \in \partial f(x).$$

Generalizes the assumption $\nabla^2 f(x) \succeq mI$ made earlier for smooth functions.

Classical Stochastic Gradient

For the finite-sum objective, get a cheap unbiased estimate of the gradient $\nabla f(x)$ by choosing an index $i \in \{1, 2, \ldots, m\}$ uniformly at random, and using $\nabla f_i(x)$ to estimate $\nabla f(x)$.

Basic SA Scheme: At iteration k, choose i_k i.i.d. uniformly at random from $\{1, 2, \ldots, m\}$, choose some $\alpha_k > 0$, and set

$$x_{k+1} = x_k - \alpha_k \nabla f_{i_k}(x_k).$$

Note that x_{k+1} depends on all random indices up to iteration k, i.e. $i_{[k]} := \{i_1, i_2, \ldots, i_k\}$.

When f is strongly convex, the analysis of convergence of expected square error $E(\|x_k - x^*\|^2)$ is fairly elementary — see Nemirovski et al (2009).

Define $a_k = \frac{1}{2}E(\|x_k - x^*\|^2)$. Assume there is $M > 0$ such that

$$\frac{1}{m}\sum_{i=1}^{m} \|\nabla f_i(x)\|_2^2 \le M.$$

Rate: $1/k$

Thus

$$\begin{aligned}
&\frac{1}{2}\|x_{k+1} - x^*\|_2^2 \\
&\quad = \frac{1}{2}\|x_k - \alpha_k \nabla f_{i_k}(x_k) - x^*\|^2 \\
&\quad = \frac{1}{2}\|x_k - x^*\|_2^2 - \alpha_k (x_k - x^*)^T \nabla f_{i_k}(x_k) + \frac{1}{2}\alpha_k^2 \|\nabla f_{i_k}(x_k)\|^2.
\end{aligned}$$

Taking expectations, get

$$a_{k+1} \le a_k - \alpha_k E[(x_k - x^*)^T \nabla f_{i_k}(x_k)] + \frac{1}{2}\alpha_k^2 M^2.$$

For middle term, have

$$\begin{aligned}
E[(x_k - x^*)^T \nabla f_{i_k}(x_k)] &= E_{i_{[k-1]}} E_{i_k}[(x_k - x^*)^T \nabla f_{i_k}(x_k) | i_{[k-1]}] \\
&= E_{i_{[k-1]}} (x_k - x^*)^T g_k,
\end{aligned}$$

... where

$$g_k := E_{i_k}[\nabla f_{i_k}(x_k) | i_{[k-1]}] \in \partial f(x_k).$$

By strong convexity, have

$$(x_k - x^*)^T g_k \geq f(x_k) - f(x^*) + \frac{1}{2} m \|x_k - x^*\|^2 \geq m \|x_k - x^*\|^2.$$

Hence by taking expectations, we get $E[(x_k - x^*)^T g_k] \geq 2 m a_k$. Then, substituting above, we obtain

$$a_{k+1} \leq (1 - 2 m \alpha_k) a_k + \frac{1}{2} \alpha_k^2 M^2.$$

When

$$\alpha_k \equiv \frac{1}{km},$$

a neat inductive argument (below) reveals the $1/k$ rate:

$$a_k \leq \frac{Q}{2k}, \qquad \text{for } Q := \max\left(\|x_1 - x^*\|^2, \frac{M^2}{m^2} \right).$$

Inductive Proof of $1/k$ Rate

Clearly true for $k = 1$. Otherwise:

$$\begin{aligned}
a_{k+1} &\le (1 - 2m\alpha_k)a_k + \frac{1}{2}\alpha_k^2 M^2 \\
&\le \left(1 - \frac{2}{k}\right) a_k + \frac{M^2}{2k^2 m^2} \\
&\le \left(1 - \frac{2}{k}\right) \frac{Q}{2k} + \frac{Q}{2k^2} \\
&= \frac{(k-1)}{2k^2} Q \\
&= \frac{k^2 - 1}{k^2} \frac{Q}{2(k+1)} \\
&\le \frac{Q}{2(k+1)},
\end{aligned}$$

as claimed.

But... What if we don't know m? Or if $m = 0$?

The choice $\alpha_k = 1/(km)$ requires strong convexity, with knowledge of the modulus m. An underestimate of m can greatly degrade the performance of the method (see example in Nemirovski et al. 2009).

Now describe a *Robust Stochastic Approximation* approach, which has a rate $1/\sqrt{k}$ (in function value convergence), and works for weakly convex nonsmooth functions and is not sensitive to choice of parameters in the step length.

This is the approach that generalizes to *mirror descent*, as discussed later.

Robust SA

At iteration k:

- set $x_{k+1} = x_k - \alpha_k \nabla f_{i_k}(x_k)$ as before;
- set
$$\bar{x}_k = \frac{\sum_{i=1}^k \alpha_i x_i}{\sum_{i=1}^k \alpha_i}.$$

For any $\theta > 0$, choose step lengths to be

$$\alpha_k = \frac{\theta}{M\sqrt{k}}.$$

Then $f(\bar{x}_k)$ converges to $f(x^*)$ in expectation with rate approximately $(\log k)/k^{1/2}$.

(The choice of θ is not critical.)

Analysis of Robust SA

The analysis is again elementary. As above (using i instead of k), have:

$$\alpha_i E[(x_i - x^*)^T g_i] \le a_i - a_{i+1} + \frac{1}{2}\alpha_i^2 M^2.$$

By convexity of f, and $g_i \in \partial f(x_i)$:

$$f(x^*) \ge f(x_i) + g_i^T (x^* - x_i),$$

thus

$$\alpha_i E[f(x_i) - f(x^*)] \le a_i - a_{i+1} + \frac{1}{2}\alpha_i^2 M^2,$$

so by summing iterates $i = 1, 2, \dots, k$, telescoping, and using $a_{k+1} > 0$:

$$\sum_{i=1}^{k} \alpha_i E[f(x_i) - f(x^*)] \le a_1 + \frac{1}{2} M^2 \sum_{i=1}^{k} \alpha_i^2.$$

Thus dividing by $\sum_{i=1} \alpha_i$:

$$E\left[\frac{\sum_{i=1}^k \alpha_i f(x_i)}{\sum_{i=1}^k \alpha_i} - f(x^*)\right] \le \frac{a_1 + \frac{1}{2}M^2 \sum_{i=1}^k \alpha_i^2}{\sum_{i=1}^k \alpha_i}.$$

By convexity, we have

$$f(\bar{x}_k) = f\left(\frac{\sum_{i=1}^k \alpha_i x_i}{\sum_{i=1}^k \alpha_i}\right) \le \frac{\sum_{i=1}^k \alpha_i f(x_i)}{\sum_{i=1}^k \alpha_i},$$

so obtain the fundamental bound:

$$E[f(\bar{x}_k) - f(x^*)] \le \frac{a_1 + \frac{1}{2}M^2 \sum_{i=1}^k \alpha_i^2}{\sum_{i=1}^k \alpha_i}.$$

By substituting $\alpha_i = \frac{\theta}{M\sqrt{i}}$, we obtain

$$\begin{aligned} E[f(\bar{x}_k) - f(x^*)] &\le \frac{a_1 + \frac{1}{2}\theta^2 \sum_{i=1}^k \frac{1}{i}}{\frac{\theta}{M}\sum_{i=1}^k \frac{1}{\sqrt{i}}} \\ &\le \frac{a_1 + \theta^2 \log(k+1)}{\frac{\theta}{M}\sqrt{k}} \\ &= M\left[\frac{a_1}{\theta} + \theta \log(k+1)\right]\frac{1}{\sqrt{k}}. \end{aligned}$$

That's it!

There are other variants — periodic restarting, averaging just over the recent iterates. These can be analyzed with the basic bound above.

Constant Step Size

We can also get rates of approximately $1/k$ for the strongly convex case, *without* performing iterate averaging. The tricks are to

- define the desired threshold ϵ for a_k in advance, and
- use a constant step size.

Recall the bound on a_{k+1} from a few slides back, and set $\alpha_k \equiv \alpha$:

$$a_{k+1} \le (1-2m\alpha)a_k + \frac{1}{2}\alpha^2 M^2.$$

Apply this recursively to get

$$a_k \le (1-2m\alpha)^k a_0 + \frac{\alpha M^2}{4m}.$$

Given $\epsilon > 0$, find α and K so that both terms on the right-hand side are less than $\epsilon/2$. The right values are:

$$\alpha := \frac{2\epsilon\mu}{M^2}, \qquad K := \frac{M^2}{4\epsilon\mu^2}\log\left(\frac{a_0}{2\epsilon}\right).$$

Constant Step Size, continued

Clearly the choice of α guarantees that the second term is less than $\epsilon/2$.

For the first term, we obtain k from an elementary argument:

$$\begin{aligned}
& (1-2m\alpha)^k a_0 \le \epsilon/2 \\
\Leftrightarrow \quad & k \log(1-2m\alpha) \le -\log(2a_0/\epsilon) \\
\Leftarrow \quad & k(-2m\alpha) \le -\log(2a_0/\epsilon) \qquad \text{since } \log(1+x) \le x \\
\Leftrightarrow \quad & k \ge \frac{1}{2m\alpha} \log(2a_0/\epsilon),
\end{aligned}$$

from which the result follows, by substituting for α in the right-hand side.

If m is underestimated by a factor of β, we undervalue α by the same factor, and K increases by $1/\beta$. (Easy modification of the analysis above.)

Thus, underestimating m gives a mild performance penalty.

Constant Step Size: Summary

PRO: Avoid averaging, $1/k$ sublinear convergence, insensitive to underestimates of m.

CON: Need to estimate probably unknown quantities: besides m, we need M (to get α) and a_0 (to get K).

We use constant size size in the parallel SG approach HOGWILD!, *to be described later.*

But the step is chosen by trying different options and seeing which seems to be converging fastest. We don't actually try to estimate all the quantities in the theory and construct α that way.

Mirror Descent

The step from x_k to x_{k+1} can be viewed as the solution of a subproblem:

$$x_{k+1} = \arg\min_z \nabla f_{i_k}(x_k)^T(z - x_k) + \frac{1}{2\alpha_k}\|z - x_k\|_2^2,$$

a linear estimate of f plus a prox-term. This provides a route to handling constrained problems, regularized problems, alternative prox-functions.

For the constrained problem $\min_{x \in \Omega} f(x)$, simply add the restriction $z \in \Omega$ to the subproblem above.

We may use other prox-functions in place of $(1/2)\|z - x\|_2^2$ above. Such alternatives may be particularly well suited to particular constraint sets Ω.

Mirror Descent is the term used for such generalizations of the SA approaches above.

Mirror Descent cont'd

Given constraint set Ω, choose a norm $\|\cdot\|$ (not necessarily Euclidean). Define the *distance-generating function* ω to be a strongly convex function on Ω with modulus 1 with respect to $\|\cdot\|$, that is,

$$(\omega'(x) - \omega'(z))^T(x - z) \geq \|x - z\|^2, \quad \text{for all } x, z \in \Omega,$$

where $\omega'(\cdot)$ denotes an element of the subdifferential.

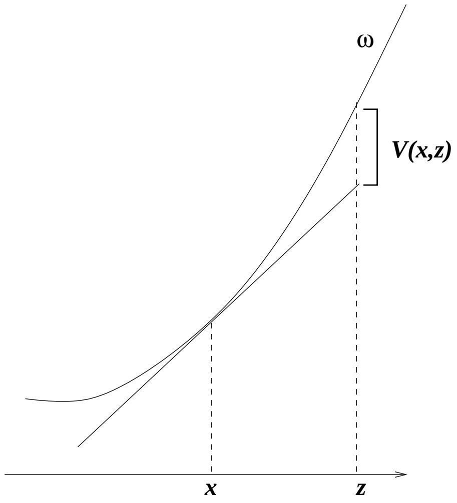
Now define the *prox-function* $V(x, z)$ as follows:

$$V(x, z) = \omega(z) - \omega(x) - \omega'(x)^T(z - x).$$

This is also known as the *Bregman distance*. We can use it in the subproblem in place of $\frac{1}{2}\|\cdot\|^2$:

$$x_{k+1} = \arg\min_{z\in\Omega} \nabla f_{i_k}(x_k)^T(z - x_k) + \frac{1}{\alpha_k}V(z, x_k).$$

Bregman distance is the deviation of ω from linearity:



Bregman Distances: Examples

For *any* Ω, we can use $\omega(x) := (1/2)\|x - \bar{x}\|_2^2$, leading to the "universal" prox-function

$$V(x,z) = (1/2)\|x - z\|_2^2$$

For the simplex

$$\Omega = \{x \in \mathbb{R}^n \,:\, x \geq 0,\, \sum_{i=1}^n x_i = 1\},$$

we can use instead the 1-norm $\|\cdot\|_1$, choose ω to be the entropy function

$$\omega(x) = \sum_{i=1}^n x_i \log x_i,$$

leading to Bregman distance (Kullback-Liebler divergence)

$$V(x,z) = \sum_{i=1}^n z_i \log(z_i/x_i),$$

which is standard measure of distance between two probability

Minibatching

$$f(x) = \frac{1}{m} \sum_{i=1}^{m} f_i(x),$$

can clump the f_i into disjoint "minibatches." Then write

$$f(x) = \frac{1}{b} \sum_{j=1}^{b} f_{[j]}(x),$$

where

$$f_{[j]}(x) = \sum_{i \in \mathcal{B}_j} f_i(x),$$

and $\mathcal{B}_1, \mathcal{B}_2, \dots, \mathcal{B}_b$ is a partition of $\{1, 2, \dots, n\}$.

Then apply SG to the batched form. Advantages:

- bigger chunks of work may make for more efficient implementation.
- lower variance in estimate of gradient $\nabla f(x)$ by $\nabla f_{[j]}(x)$.
- allows parallelism — the work of evaluating $\nabla f_i(x)$ for $i \in \mathcal{B}_j$ can be farmed out to various processors or cores.

Parallel Stochastic Gradient

Several approaches tried, for $f(x) = \sum_{i=1}^{m} f_i(x)$.

- **Asynchronous:** HOGWILD!**:** Each core grabs the centrally-stored x and evaluates $\nabla f_i(x)$ for some random i, then writes the updates back into x (Niu, Ré, Recht, Wright, NIPS, 2011).
- **Synchronous:** "Internal" parallelism in evaluation of gradient $\nabla f_{[j]}(x)$ for minibatch $\mathcal{B}_j$. (No changes needed to analysis.)

HOGWILD!: Each processor runs independently:

1. Sample i uniformly from $\{1, 2, \ldots, m\}$;
2. Read current state of x and evaluate $g_i = \nabla f_i(x)$;
3. Update $x \leftarrow x - \alpha g_i$;

Hogwild! Convergence

Analysis of asynchronous methods is nontrivial

- Updates can be old by the time they are applied, but we assume a bound τ on their age. This value τ is related to the number of cores involved in the computation.
- Processors can overwrite each other's work, but this effect is less important if each f_i depends on just a few components of x

Analysis of Niu et al (2011) simplified / generalized by Richtarik (2012) and many others.

Essentially show that similar $1/k$ rate to serial SG are possible provided that the maximum age τ of the updates is not too large.

Coordinate Descent

Consider unconstrained smooth problem: $\min f(x)$. In coordinate descent (CD), we take a step in one component of x at a time.

```
Set Choose $x^1 \in \mathbb{R}^n$;
for $\ell = 0, 1, 2, \ldots$ do
    for $j = 1, 2, \ldots, n$ do
        Define $k = \ell n + j$;
        Choose index $i = i(\ell, j) \in \{1, 2, \ldots, n\}$;
        Choose $\alpha_k > 0$;
        $x^{k+1} \leftarrow x^k - \alpha_k \nabla_i f(x^k) e_i$;
    end for
end for
```

Here $e_i = (0, \ldots, 0, 1, 0, \ldots, 0)^T$, with the 1 in position i.

Each ℓ represents one "epoch" (batch of n CD steps). Each j is an inner iteration.

Allows generalizations to *blocks* (multiple components), separable regularization terms.

Economics of CD

CD is most useful when the cost of one step is about $1/n$ of the cost of a full gradient step. These "economics" hold true in some important classes of problems:

- Empirical Risk Minimization (ERM):

$$f(x) = \frac{1}{m} \sum_{j=1}^{m} h_j(A_{j\cdot} x) + \lambda \sum_{i=1}^{n} \Omega_i(x_i),$$

  where $A_{j\cdot}$ is j-th row of an $m \times n$ matrix A. Examples: Least squares, logistic regression, support vector machines.

- Graph-based objectives:

$$f(x) = \sum_{(i,j) \in E} f_{ij}(x_i, x_j),$$

  where each variable x_i is associated with a node in the graph and E is the set of edges.

On certain problems, CD can be faster than full-gradient steepest descent — up to n times faster.

CD Variants

One way to distinguish between different variants of CD is by the order in which components are considered. That is, the choice of $i(\ell, j)$ at epoch ℓ, inner iteration j.

- CCD (Cyclic CD): $i(\ell, j) = j$.
- RCD (Randomized CD a.k.a. Stochastic CD): $i(\ell, j)$ is chosen uniformly at random from $\{1, 2, \dots, n\}$ — sampling-with-replacement.
- RPCD (Random-Permutations Cyclic CD): At the start of epoch ℓ, we choose a random permutation of $\{1, 2, \dots, n\}$, denoted by π_ℓ. Index $i(\ell, j)$ is chosen to be the jth entry in π_ℓ. This is sampling without replacement within each cycle.

Choice of α_k is also important — exact line search, or some other, looser step along $\nabla_i f(x^k)$.

Important Quantities

We assume convex f. Certain global properties are important for convergence analysis.

$L_{\max}$ is a componentwise Lipschitz constant for ∇f:

$$|\nabla_i f(x + te_i) - \nabla_i f(x)| \leq L_i |t|, \quad L_{\max} = \max_{i=1,2,\dots,n} L_i.$$

L is the Lipschitz constant for the whole gradient:

$$\|\nabla f(y) - \nabla f(z)\| \leq L\|y - z\|.$$

On quadratic functions, we have

$$1 \leq \frac{L}{L_{\max}} \leq n.$$

The max is achieved by $f(x) = (e^T x)^2$, where $e = (1, 1, \dots, 1)^T$

RCD Convergence

Convergence of sampling-with-replacement randomized with $\alpha_k = 1/L_{\max}$.

$$\begin{aligned}
f(x^{k+1}) &= f\left(x^k - \alpha_k \nabla_{i_k} f(x^k) e_{i_k}\right) \\
&\le f(x^k) - \alpha_k [\nabla_{i_k} f(x^k)]^2 + \frac{1}{2}\alpha_k^2 L_{i_k} [\nabla_{i_k} f(x^k)]^2 \\
&\le f(x^k) - \alpha_k \left(1 - \frac{L_{\max}}{2}\alpha_k\right) [\nabla_{i_k} f(x^k)]^2 \\
&= f(x^k) - \frac{1}{2L_{\max}} [\nabla_{i_k} f(x^k)]^2,
\end{aligned}$$

where we used $\alpha_k = 1/L_{\max}$. Take expectations w.r.t. i_k (note that x^k does not depend on i_k), get

$$\begin{aligned}
E_{i_k}[f(x^{k+1})] &\le f(x^k) - \frac{1}{2L_{\max}} \frac{1}{n} \sum_{i=1}^{n} [\nabla_i f(x^k)]^2 \\
&= f(x^k) - \frac{1}{2nL_{\max}} \|\nabla f(x^k)\|^2.
\end{aligned}$$

Subtract f^* from both sides and take expectation w.r.t $i_0, i_1, \dots, i_{k-1}$:

$$\phi_{k+1} \le \phi_k - \frac{1}{2nL_{\max}} E\left(\|\nabla f(x^k)\|^2\right).$$

If f is strongly convex with modulus μ, we have

$$f(y) \ge f(x) + \nabla f(x)^T(y-x) + \frac{\mu}{2}\|y-x\|^2.$$

Fixing $x = x^k$ and minimizing both sides of this expression (separately) w.r.t. y, obtain

$$f^* \ge f(x^k) - \frac{1}{2\mu}\|\nabla f(x^k)\|^2$$

so by rearranging and taking expectations:

$$E\left(\|\nabla f(x^k)\|^2\right) \ge 2\mu E[f(x^k) - f^*] = 2\mu\phi_k.$$

Thus get linear convergence in expectation:

$$\phi_{k+1} \le \left(1 - \frac{\mu}{nL_{\max}}\right)\phi_k, \quad k = 0, 1, 2, \dots .$$

Nesterov (2012)

Comparison with Steepest Descent

This rate is for a single CD step. Over an epoch of n steps, we get a decrease of

$$\left(1 - \frac{\mu}{nL_{\max}}\right)^n \approx \left(1 - \frac{\mu}{L_{\max}}\right).$$

Compare this with the decrease factor for one step of steepest descent (SD) with a fixed step $\alpha_k = 1/L$ (analyzed earlier):

$$\left(1 - \frac{\mu}{L}\right).$$

When $L_{\max} \ll L$, RCD can be a lot faster than SD!

To get convergence to $\phi_K \leq \epsilon$, need approximately

$$\begin{aligned} \frac{L}{\mu}|\log \epsilon| &\quad \text{SD iterations} \\ \frac{L_{\max}}{\mu}|\log \epsilon| &\quad \text{RCD epochs.} \end{aligned}$$

Cyclic CD Convergence

Convergence rate results for CCD are somewhat weaker (Beck and Tetruashvili (2013)). Depending on choice of steplength, the worst-case bounds are generally worse than for RCD or SD.

$$\begin{aligned}
&\text{CCD with } \alpha = 1/L_{\max}: && \frac{2nL^2/L_{\max}}{\mu}|\log\epsilon| \\
&\text{CCD with } \alpha = 1/L: && \frac{2Ln}{\mu}|\log\epsilon| \\
&\text{CCD with } \alpha = 1/(\sqrt{n}L): && \frac{4L\sqrt{n}}{\mu}|\log\epsilon|.
\end{aligned}$$

- "Worse" than the expected linear rate for RCD by factors of $\sqrt{n}L/L_{\max}$ to $nL^2/L_{\max}^2$;
- "Worse" than linear rate for SD by factors of $\sqrt{n}$ to $nL/L_{\max}$.

CD Convergence

CCD can achieve this worst-case behavior for a particular quadratic function, with Hessian

$A = (1-\delta)ee^T + \delta I, \quad$ where $e = (1, 1, \dots, 1)^T$ and δ is small and positive.

(see Sun and Ye (2016).)

Random-Permutations Cyclic (RPCD) behaves provably well on this example (Lee and Wright (2016)) — better than RCD.

In general RPCD works as well (or better than) RCD, but it is difficult to analyze in general.

Still, on many problems, when $L/L_{\max}$ is moderate, there is not much difference in performance between SD and all variants of CD.

References I

Beck, A. and Tetruashvili, L. (2013). On the convergence of block coordinate descent type methods. *SIAM Journal on Optimization*, 23(4):2037–2060.

Lee, C.-p. and Wright, S. J. (2016). Random permutations fix a worst case for cyclic coordinate descent. Technical Report arXiv:1607.08320, Computer Sciences Department, University of Wisconsin-Madison.

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

Sun, R. and Ye, Y. (2016). Worst-case complexity of cyclic coordinate descent: $o(n^2)$ gap with randomized version. Technical Report arXiv:1604.07130, Department of Management Science and Engineering, Stanford University, Stanford, California.

Wright, S. J. (2015). Coordinate descent algorithms. *Mathematical Programming, Series B*, 151:3–34.