6
Coordinate Descent

Coordinate descent (CD) methods minimize a multivariate function by changing one of the variables (or sometimes a “block” of variables) to decrease the objective function while holding the others fixed. Such methods have a certain intuitive appeal, as they replace the multivariate optimization problem by a sequence of scalar (or lower-dimensional) problems, for which steps can be taken more cheaply. There are many variants and extensions of the basic CD approach that have gone in and out of style over the years. The latest wave of interest is driven largely by the usefulness of CD methods in machine learning and data analysis problems.

To describe the approach, we focus on the basic method in which a single coordinate is chosen for updating at each iteration of coordinate descent. When applied to a function $f\colon \mathbb{R}^n \to \mathbb{R}$, the kth iteration chooses some index $i_k \in \{1, 2, \ldots, n\}$ and takes a step of the form

$$x^{k+1} \leftarrow x^k + \gamma_k e_{i_k}, \tag{6.1}$$

where e_{i_k} is the i_k unit vector and γ_k is the step. In one variant of CD (also known as the Gauss–Seidel method), γ_k is chosen to minimize f along direction e_{i_k}:

$$\gamma_k := \arg\min_{\gamma} f(x^k + \gamma e_{i_k}).$$

More practical variants do not minimize exactly along the coordinate directions but rather choose γ_k to be a negative multiple of the partial derivative $\partial f / \partial x_{i_k}$ (also denoted by $\nabla_{i_k} f$):

$$x^{k+1} \leftarrow x^k - \alpha_k \nabla_{i_k} f(x^k) e_{i_k}, \tag{6.2}$$

for some $\alpha_k > 0$. Different variants of CD are distinguished by different techniques for choosing i_k and α_k. In this chapter, we focus mainly on methods

https://doi.org/10.1017/9781009004282.007 Published online by Cambridge University Press

of type (6.2) with fixed values of α_k that are defined in terms of Lipschitz constants for the gradients, as for the full gradient methods of Section 3.2.

Section 6.1 illustrates two important optimization formulations in machine learning in which the per-iteration cost of CD is much lower (possibly by a factor of n) than the per-iteration cost of a full gradient method, making CD an potentially competitive approach. In Section 6.2, we describe complexity results for CD applied to convex functions for two variants of CD. The worst-case analysis for one of these approaches – the one in which the index i_k is chosen randomly and independently of previous iterations for all k – can be stronger than that of full gradient descent, when factor-of-n per-iteration savings are realized for CD iterations. (Section 9.4 extends the result for randomized CD to functions that are strongly convex and that contain separable convex regularization terms.) Practical variants of CD often take steps in *blocks* of variables at a time rather than in a single variable. The analysis of such cases is not vastly different from single-variable CD, and we discuss these block-CD variants in Section 6.3.

6.1 Coordinate Descent in Machine Learning

In deciding whether CD is a plausible approach for minimizing f, relative to such alternatives as the gradient methods of Chapters 3 and 4, we need to consider how the properties and structure of f impact the economics of the approach. Since CD typically requires more steps than full gradient methods, they make sense only if the cost of computing them is correspondingly lower. That is, the cost of computing partial gradient information needs to be cheaper than computing the full gradient, and the computation and bookkeeping required to take the step also should be relatively inexpensive. We describe two examples from machine learning in which these properties hold, making them good candidates for CD.

Coordinate Descent for Empirical Risk Minimization. Consider the objective that arises in regularized regression, classification, and ERM problems:

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

where each ϕ_j is a convex loss, $A_{j\cdot}$ denotes the jth row of the $N \times n$ matrix A, the functions Ω_i, $i = 1, 2, \ldots, n$ are convex regularization functions, and $\lambda \geq 0$ is a regularization parameter. (We assume for the present that the functions ϕ_j

and Ω_i are all differentiable) Although computing the ith component of the gradient ($\nabla_i f$) is expensive, it is easy to store and update information to lower this cost greatly. The trick is to store the vector $g = Ax$ for the current x, along with the scalars $\nabla\phi_j(g_j)$, $j = 1,2,\dots,N$. We then have

$$\nabla_i f(x) = \frac{1}{N}\sum_{j=1}^{N} A_{j,i}\nabla\phi_j(g_j) + \lambda\nabla\Omega_i(x_i),$$

where $A_{j,i}$ denotes the (j,i) element of the matrix A. Note that the terms in the summation need be evaluated only for those indices j for which $A_{j,i}$ is nonzero; that is,

$$\nabla_i f(x) = \frac{1}{N}\sum_{j:A_{j,i}\neq 0} A_{j,i}\nabla\phi_j(g_j) + \lambda\nabla\Omega_i(x_i).$$

This computation costs $O(|A_{\cdot i}|)$ operations, where $A_{\cdot i}$ is the ith column of A. (The number of operations required to compute the full gradient would be proportional to the number of nonzeros in the full matrix A.) Additionally, the cost of updating the quantities $g_j := A_{j\cdot}x$ and $\nabla\phi_j(g_j)$, $j = 1,2,\dots,N$ following a step γ_i along the coordinate direction x_i is also reasonable. The update formulas for the components of g are

$$g_j \leftarrow g_j + A_{j,i}\gamma_i, \quad j = 1,2,\dots,N,$$

so it is necessary to update only those g_j (and $\nabla\phi_j(g_j)$) for which $A_{j,i} \neq 0$ – a total workload of $O(|A_{\cdot i}|)$ operations. Considering all possible choices of components $i = 1,2,\dots,n$, we see that the expected cost per iteration of CD is about $O(|A|/n)$, where $|A|$ is the number of nonzeros in A. The cost per iteration of a gradient method would be $O(|A|)$. This is a large advantage for CD methods – a factor of $1/n$ – that makes CD potentially appealing relative to full gradient methods.

The least-squares problem $\min \frac{1}{2N}\|A^T x - b\|_2^2$ is a special case of this example, as we see by defining $\phi_j(g_j) = \frac{1}{2}(g_j - b_j)^2$.

Graph-Structured Objectives. Many optimization can be written as a sum of functions, each of which involves only two components of the vector of variables. For example, problems in image segmentation might couple pixels only when they are adjacent. In topic modeling, terms may be coupled only when they appear in the same document.

We can express the structure of such a function as an undirected graph $G = (V,E)$, where each edge $(j,l) \in E$ connects two vertices j and l from $V = \{1,2,\dots,n\}$. The objective has the form

$$f(x) = \sum_{(j,l)\in E} f_{jl}(x_j, x_l) + \lambda \sum_{j=1}^{n} \Omega_j(x_j).$$

(We assume that each f_{jl} and each regularization function Ω_j is differentiable.) If we assume that evaluation of each gradient ∇f_{jl} and $\nabla \Omega_j$ is an $O(1)$ operation, the cost of a full gradient ∇f would be $O(|E|+n)$. To implement a CD method efficiently, we would store the values of f_{jl} and ∇f_{jl} at the current x, for all $(j,l) \in E$. To compute the ith gradient component $\nabla_i f(x)$, we need to sum components from the terms $\nabla f_{jl}(x)$ for which $j = i$ or $l = i$ (at a total cost proportional to the number of edges incident on vertex i) and evaluate the term $\nabla \Omega_i(x_i)$. In updating the values of f_{jl} and ∇f_{jl} after the step in x_i, we need again only change those components for which $j = i$ or $l = i$. The "expected" cost of one CD iteration is thus $O(|E|/n)$. We see once again the desired $1/n$ relationship between the cost per iteration of CD and the cost per iteration of a gradient method.

In both of these cases, the amount of computation required to update f is similar to that required to update the full gradient vector, and some of the actual operations are the same (for example, the update of g_j terms in the ERM example). This observation suggests that we can perform line searches along the coordinate directions efficiently, using information about changes in function and directional derivative information to find near-exact minima along each search direction.

If we are using a naive finite difference scheme to estimate derivatives, based on a formula such as

$$\nabla_i f(x) \approx \frac{f(x+\delta e_i) - f(x)}{\delta},$$

then n function evaluations are required to evaluate a full gradient, compared with 1 to estimate a single component. However, we note in this connection that automatic differentiation techniques (Griewank and Walther, 2008), implemented in many software packages, can compute ∇f for a modest multiple (independent of n) of the cost of evaluating f. (Note that this observation is not really relevant to the examples of this section, since for these objectives, the cost of evaluating f is itself too high.)

6.2 Coordinate Descent for Smooth Convex Functions

We again develop most of the ideas with reference to the familiar smooth convex minimization problem defined by

$$\min_{x\in\mathbb{R}^n} f(x), \tag{6.3}$$

where f is smooth and convex, with modulus of convexity m and a bound L on the Lipschitz constant of the gradient for all points x in some region of interest; see (2.19) and (2.7). We showed in Lemmas 2.3 and 2.9 that, in the case of f twice continuously differentiable, these conditions are a consequence of uniform bounds on the eigenvalues of the Hessian (2.10) – that is, $mI \preceq \nabla^2 f(x) \preceq LI$. Because the variants we consider here are mostly descent methods, it is enough to restrict our attention in these definitions to an open neighborhood $\mathcal{O}^0$ of the level set of f for the starting point x^0, which is $\mathcal{L}^0 := \{x \mid f(x) \le f(x^0)\}$.

6.2.1 Lipschitz Constants

We introduce other partial Lipschitz constants for the gradient ∇f. Each *componentwise Lipschitz constant* L_i, $i = 1, 2, \dots, n$ satisfies the bound

$$|\nabla_i f(x + \gamma e_i) - \nabla_i f(x)| \le L_i |\gamma|, \quad i = 1, 2, \dots, n, \tag{6.4}$$

for all x, γ such that $x \in \mathcal{O}^0$ and $x + \gamma e_i \in \mathcal{O}^0$, while we define $L_{\max}$ to be the maximum of these constants:

$$L_{\max} := \max_{i=1,2,\dots,n} L_i. \tag{6.5}$$

These Lipschitz constants play important roles both in implementing variants of CD and in analyzing its convergence rates and in comparing these rates with those of full gradient methods. We can obtain some bounds on the difference between L and $L_{\max}$ by considering the convex quadratic function $f(x) = (1/2)x^T Ax$ where A is symmetric positive semidefinite. We have that

$$L = \|A\|_2 = \lambda_{\max}(A), \quad L_{\max} = \max_{i=1,2,\dots,n} A_{ii}.$$

It is clear from definition of matrix norm that

$$L \ge \|Ae_i\| / \|e_i\| = \sqrt{\sum_{j=1}^{n} A_{ji}^2} \ge A_{ii},$$

from which it follows that $L \ge L_{\max}$. (Equality holds for any nonnegative diagonal matrix.) On the other hand, we have by the relationship between trace and sum of eigenvalues (A.4) that

$$L = \lambda_{\max}(A) \le \sum_{i=1}^{n} \lambda_i(A) = \sum_{i=1}^{n} A_{ii} \le nL_{\max}.$$

(Equality holds for the matrix $A = ee^T$, where $e = (1, 1, \dots, 1)^T$.) Thus, we have

$$L_{\max} \le L \le nL_{\max}. \tag{6.6}$$

6.2.2 Randomized CD: Sampling with Replacement

In the basic randomized coordinate descent (RCD) approach, the index i_k to be updated is selected uniformly at random from $\{1, 2, \dots, n\}$, and the iterations have the form (6.2) for some $\alpha_k > 0$. For short-step variants, in which α_k is determined by the Lipschitz constants rather than by exact minimization or a line-search process, sublinear convergence rates can be attained for convex functions and linear convergence rates for strongly convex functions ($m > 0$ in (2.19)). Later, we discuss how this rate relates to the rates obtained for the full gradient steepest-descent method of Chapter 3.

For precision, we make the following assumption for the remainder of this section. We make use here of the level set $\mathcal{L}^0$ and its open neighborhood $\mathcal{O}^0$ defined earlier.

Assumption 1 The function f is convex and uniformly Lipschitz continuously differentiable on the set $\mathcal{O}^0$ defined earlier, and attains its minimum on a set $\mathcal{S} \subset \mathcal{L}^0$. There is a finite $R_0 > 0$ for which the following bound is satisfied:

$$\max_{x \in \mathcal{L}^0} \min_{x^* \in \mathcal{S}} \|x - x^*\| \le R_0.$$

In the analysis that follows, we denote expectation with respect to a single random index i_k by $\mathbb{E}_{i_k}(\cdot)$, while $\mathbb{E}(\cdot)$ denotes expectation with respect to all random variables $i_0, i_1, i_2, \dots$ encountered during the algorithm.

Our main result shows convergence of randomized CD for the fixed steplength $\alpha_k \equiv 1/L_{\max}$.

Theorem 6.1 *Suppose that Assumption 1 holds, that each index i_k in the iteration* (6.2) *is selected uniformly at random from* $\{1, 2, \dots, n\}$, *and that $\alpha_k \equiv 1/L_{\max}$. Then for all $k > 0$, we have*

$$\mathbb{E}(f(x^k)) - f^* \le \frac{2nL_{\max}R_0^2}{k}. \tag{6.7}$$

When $m > 0$ in (2.19), *we have, in addition, that*

$$\mathbb{E}\left(f(x^k)\right) - f^* \le \left(1 - \frac{m}{nL_{\max}}\right)^k (f(x^0) - f^*). \tag{6.8}$$

Proof By application of Taylor's theorem, and using (6.4) and (6.5), we have

$$\begin{aligned} f(x^{k+1}) &= f\left(x^k - \alpha_k \nabla_{i_k} f(x^k) e_{i_k}\right) \\ &\le f(x^k) - \alpha_k [\nabla_{i_k} f(x^k)]^2 + \frac{1}{2}\alpha_k^2 L_{i_k} [\nabla_{i_k} f(x^k)]^2 \\ &\le f(x^k) - \alpha_k \left(1 - \frac{L_{\max}}{2}\alpha_k\right) [\nabla_{i_k} f(x^k)]^2 \\ &= f(x^k) - \frac{1}{2L_{\max}} [\nabla_{i_k} f(x^k)]^2, \end{aligned} \tag{6.9}$$

where we substituted the choice $\alpha_k = 1/L_{\max}$ in the last equality. Taking the expectation of both sides of this expression over the random index i_k, we have

$$\begin{aligned} \mathbb{E}_{i_k} f(x^{k+1}) &\le f(x^k) - \frac{1}{2L_{\max}} \frac{1}{n} \sum_{i=1}^n [\nabla_i f(x^k)]^2 \\ &= f(x^k) - \frac{1}{2nL_{\max}} \|\nabla f(x^k)\|^2. \end{aligned} \tag{6.10}$$

(We used here the facts that x^k does not depend on i_k and that i_k was chosen from among $\{1, 2, \dots, n\}$ with equal probability.) We now subtract $f(x^*)$ from both sides of this expression and take expectation of both sides with respect to *all* random variables $i_0, i_1, \dots$, using the notation

$$\phi_k := \mathbb{E}(f(x^k)) - f^*, \tag{6.11}$$

to obtain

$$\phi_{k+1} \le \phi_k - \frac{1}{2nL_{\max}} \mathbb{E}\left(\|\nabla f(x^k)\|^2\right) \le \phi_k - \frac{1}{2nL_{\max}} \left[\mathbb{E}(\|\nabla f(x^k)\|)\right]^2. \tag{6.12}$$

(We used Jensen's inequality in the second inequality.) We see already from this last inequality that $\{\phi_k\}$ is a nonincreasing sequence. By convexity of f, we have for any $x^* \in \mathcal{S}$ that

$$f(x^k) - f^* \le \nabla f(x^k)^T (x^k - x^*) \le \|\nabla f(x^k)\| \|x^k - x^*\| \le R_0 \|\nabla f(x^k)\|,$$

where the final inequality is obtained from Assumption 1, because $f(x^k) \le f(x^0)$, so that $x^k \in \mathcal{L}^0$. By taking expectations of both sides, we have

$$\mathbb{E}(\|\nabla f(x^k)\|) \ge \frac{1}{R_0}\phi_k.$$

When we substitute this bound into (6.12) and rearrange, we obtain

$$\phi_k - \phi_{k+1} \ge \frac{1}{2nL_{\max}} \frac{1}{R_0^2} \phi_k^2.$$

We thus have

$$\frac{1}{\phi_{k+1}} - \frac{1}{\phi_k} = \frac{\phi_k - \phi_{k+1}}{\phi_k \phi_{k+1}} \geq \frac{\phi_k - \phi_{k+1}}{\phi_k^2} \geq \frac{1}{2nL_{\max} R_0^2}.$$

By applying this formula recursively, we obtain

$$\frac{1}{\phi_k} \geq \frac{1}{\phi_0} + \frac{k}{2nL_{\max} R_0^2} \geq \frac{k}{2nL_{\max} R_0^2},$$

from which (6.7) follows.

In the case of f strongly convex with modulus $m > 0$, we have by taking the minimum of both sides with respect to y in (2.19), and setting $x = x^k$, that

$$f^* \geq f(x^k) - \frac{1}{2m} \|\nabla f(x^k)\|^2.$$

By using this expression to bound $\|\nabla f(x^k)\|^2$ in (6.12), we obtain

$$\phi_{k+1} \leq \phi_k - \frac{m}{nL_{\max}} \phi_k = \left(1 - \frac{m}{nL_{\max}}\right) \phi_k.$$

Recursive application of this formula leads to (6.8). □

The same convergence expressions can be obtained for more refined choices of steplength α_k by making minor adjustments to the logic in (6.9). For example, the (usually longer) steplength $\alpha_k = 1/L_{i_k}$ leads to the same bounds (6.7) and (6.8). The same bounds hold too when α_k is the exact minimizer of f along the coordinate search direction; we modify the logic in (6.9) for this case by taking the minimum of all expressions with respect to α_k and use the fact that $\alpha_k = 1/L_{\max}$ is, in general, a suboptimal choice.

We prove a second convergence result, with a bound different from (6.7), for the weakly convex case. This variant, from Lu and Xiao (2015, theorem 1), is also of interest because it uses a different proof technique.

Theorem 6.2 *Suppose that Assumption 1 holds, that each index i_k in the iteration* (6.2) *is selected uniformly at random from* $\{1, 2, \ldots, n\}$, *and that* $\alpha_k \equiv 1/L_{\max}$. *Then for all $k > 0$, we have*

$$\mathbb{E}(f(x^k)) - f^* \leq \frac{nL_{\max} R_0^2}{2k} + \frac{n(f(x^0) - f(x^*))}{k} \leq \frac{n(L_{\max} + L) R_0^2}{2k}. \tag{6.13}$$

Proof Define ϕ_k as in (6.11) and

$$a_k := \mathbb{E}(\|x^k - x^*\|^2) \tag{6.14}$$

for some minimizer x^* of f, where the expectation $\mathbb{E}$ is taken over all random indices $i_0, i_1, \ldots$. For any iteration T, we have

$$\begin{aligned}
&\|x^{T+1} - x^*\|^2 \\
&\quad = \left\| x^T - \frac{1}{L_{\max}} \nabla_{i_T} f(x^T) e_{i_T} - x^* \right\|^2 \\
&\quad = \|x^T - x^*\|^2 - \frac{2}{L_{\max}} \nabla_{i_T} f(x^T)(x^T - x^*)_{i_T} + \frac{1}{L_{\max}^2} \left[\nabla_{i_T} f(x^T) \right]^2 \\
&\quad \le \|x^T - x^*\|^2 - \frac{2}{L_{\max}} \nabla_{i_T} f(x^T)(x^T - x^*)_{i_T} + \frac{2}{L_{\max}} \left[f(x^T) - f(x^{T+1}) \right],
\end{aligned}$$

where the last inequality is obtained by applying (6.9) to the last term. By taking the expectations of both sides with respect to the random index i_T, and using the fact that $f(x^*) \ge f(x^T) + \nabla f(x^T)^T (x^* - x^T)$ (by convexity), we have

$$\begin{aligned}
\mathbb{E}_{i_T} \|x^{T+1} - x^*\|^2 &\le \|x^T - x^*\|^2 - \frac{2}{nL_{\max}} \nabla f(x^T)^T (x^T - x^*) \\
&\quad + \frac{2}{L_{\max}} \left[f(x^T) - \mathbb{E}_{i_T} f(x^{T+1}) \right] \\
&\le \|x^T - x^*\|^2 + \frac{2}{nL_{\max}} (f(x^*) - f(x^T)) \\
&\quad + \frac{2}{L_{\max}} \left[f(x^T) - \mathbb{E}_{i_T} f(x^{T+1}) \right],
\end{aligned}$$

which, by rearrangement, yields

$$\begin{aligned}
\frac{2}{nL_{\max}} (f(x^T) - f(x^*)) &\le \|x^T - x^*\|^2 - \mathbb{E}_{i_T} \|x^{T+1} - x^*\|^2 \\
&\quad + \frac{2}{L_{\max}} [f(x^T) - \mathbb{E}_{i_T} f(x^{T+1})].
\end{aligned}$$

By taking expectations of both sides over all random indices $i_0, i_1, \ldots$, and using the definitions (6.11) and (6.14), we obtain

$$\frac{2}{nL_{\max}} \phi_T \le a_T - a_{T+1} + \frac{2}{L_{\max}} (\phi_T - \phi_{T+1}).$$

By summing both sides over $T = 0, 1, \ldots, k$, we obtain

$$\begin{aligned}
\frac{2}{nL_{\max}} \sum_{T=0}^{k} \phi_T &\le a_0 - a_{k+1} + \frac{2}{L_{\max}} (\phi_0 - \phi_{k+1}) \\
&\le \|x^0 - x^*\|^2 + \frac{2[f(x^0) - f(x^*)]}{L_{\max}}, \qquad (6.15)
\end{aligned}$$

where for the last inequality, we used $a_0 = \|x^0 - x^*\|^2$ and $\phi_0 = f(x^0) - f(x^*)$ along with $a_{k+1} \geq 0$ and $\phi_{k+1} \geq 0$. Since $\{f(x^T)\}$ is a monotonically decreasing sequence, we can bound the left-hand side of (6.15) below by $(k+1)\frac{2}{nL_{\max}}\phi_k$, and by substituting this expression into (6.15), we obtain

$$\phi_k = \mathbb{E} f(x^k) - f^* \leq \frac{nL_{\max}\|x^0 - x^*\|^2}{2(k+1)} + \frac{n(f(x^0) - f^*)}{k+1},$$

and we simply replace $k+1$ with k on the right-hand side to obtain the result.

The final bound in the theorem is obtained by using convexity and $\nabla f(x^*) = 0$ to obtain

$$f(x^0) \leq f(x^*) + \nabla f(x^*)^T(x^0 - x^*) + \frac{L}{2}\|x^0 - x^*\|^2 = f(x^*) + \frac{L}{2}\|x^0 - x^*\|^2.$$

□

The convergence rates in Theorem 6.1 make interesting comparisons with the corresponding rates for full gradient short-step methods from Section 3.2. In comparing (6.7) with the corresponding result for the (full gradient) steepest-descent method with constant steplength $\alpha_k = 1/L$ (where L is from (2.7)). We showed in Theorem 3.3 that the iteration

$$x^{k+1} = x^k - \frac{1}{L}\nabla f(x^k)$$

leads to a convergence expression

$$f(x^k) - f^* \leq \frac{LR_0^2}{2k}. \tag{6.16}$$

Since, for problems of interest in this chapter, there is roughly a factor-of-n difference between one iteration of CD and one iteration of a full gradient method, the bounds (6.16) and (6.7) would be comparable (to within a factor of 4) if L and $L_{\max}$ are approximately the same. The bounds (6.6) suggest that $L_{\max}$ can be significantly less than L for some problems, and by comparing the two worst-case convergence expressions, we see that randomized CD may have an advantage in such cases.

A similar conclusion is reached when we compare the convergence rates on the strongly convex case. We have for the steepest-descent method with line search $\alpha \equiv 2/(L+m)$ (see Section 3.2) that

$$\|x_{k+1} - x^*\| \leq \left(1 - \frac{2}{(L/m)+1}\right)\|x_k - x^*\|. \tag{6.17}$$

Because of Lemma 3.4, the quantities $f(x_k) - f(x^*)$ and $\|x_k - x^*\|^2$ converge at similar rates, so we get a more apt comparison with (6.8) by squaring both sides of (6.17). By using the approximation $(1-\epsilon)^r \approx 1-r\epsilon$ for any constants r and ϵ with $r\epsilon \ll 1$, we estimate that the rate constant for convergence of $\{f(x_k)\}$ in the short-step steepest-descent method would be about

$$1 - \frac{4m}{L+m} \approx 1 - \frac{4m}{L}, \tag{6.18}$$

because we can assume that $L + m \approx L$ for all but the most well-conditioned problems. Apart from the extra factor of 4 in (6.18), and the expected factor-of-n difference between the key terms, we note again that the main difference is the replacement of $L_{\max}$ in (6.8) by L in (6.18). Again, we note the possibility of a faster overall rate for CD when $L_{\max}$ is significantly less than L.

These observations make intuitive sense. CD methods are able to take longer steps in general while still guaranteeing significant decrease in f. Moreover, they make incremental improvements to x using fresh gradient information at every step, whereas full gradient methods update all components of x at once using information from all components of the gradient at a single point.

Complexity results for the case of strongly convex f appear in Section 9.4. In fact, we consider there the more general situation in which convex separable regularization functions are added to f, and a proximal-gradient framework (which generalizes gradient descent to regularized objective functions) is used to minimize them.

6.2.3 Cyclic CD

The cyclic variant of CD updates the coordinates in sequential order $1, 2, \dots, n$, then repeats the cycle until convergence is declared. This is perhaps the most intuitive form of the algorithm. The classical Gauss–Seidel method, popular also for linear systems of equations, has this form, with the steplengths chosen to minimize f exactly along each search direction. Other variants do not minimize exactly but rather take steps of the form (6.2), with α_k chosen according to estimates of the Lipschitz properties of the function, and other considerations.

In the general CD framework (6.1), the choice of index i_k in cyclic CD is

$$i_k = (k \bmod n) + 1, \quad k = 0, 1, 2, \dots, \tag{6.19}$$

giving the sequence $1, 2, 3, \dots, n, 1, 2, 3, \dots, n, 1, 2, 3, \dots$.

Surprisingly, results concerning the convergence of cyclic variants for smooth convex f have emerged only recently. See, for example, Beck and Tetruashvili (2013) from which the results below are extracted; Sun and Hong

(2015); and Li et al. (2018). (Results for the special case of the Gauss–Seidel method applied to a convex quadratic, f, and its important symmetric over-relaxation (SOR) variant, have been standard results in numerical linear algebra for many years.) We describe a result with a flavor similar to Theorem 6.1, assuming a fixed steplength α at every iteration, where $\alpha \leq 1/L_{\max}$.

Theorem 6.3 *Suppose that Assumption 1 holds and that the iteration* (6.2) *is applied with the index i_k at iteration k chosen according to the cyclic ordering* (6.19) *and $\alpha_k \equiv \alpha \leq 1/L_{\max}$. Then, for $k = n, 2n, 3n, \dots$, we have*

$$f(x^k) - f^* \leq \frac{(4n/\alpha)(1 + nL^2\alpha^2)R_0^2}{k+8}. \tag{6.20}$$

When f is strongly convex with modulus m, we have, in addition, for $k = n, 2n, 3n, \dots$, that

$$f(x^k) - f^* \leq \left(1 - \frac{m}{(2/\alpha)(1 + nL^2\alpha^2)}\right)^{k/n} (f(x^0) - f^*). \tag{6.21}$$

Proof The results follow from Beck and Tetruashvili (2013, theorems 3.6 and 3.9). We note that (i) each iteration of Algorithm BCGD in Beck and Tetruashvili (2013) corresponds to a "cycle" of n iterations of (6.2); (ii) we update coordinates rather than blocks, so that the parameter p in Beck and Tetruashvili (2013) is equal to n; (iii) we set $\bar{L}_{\max}$ and $\bar{L}_{\min}$ in Beck and Tetruashvili (2013) both to $1/\alpha$, which is greater than or equal to $L_{\max}$, as required by the proofs in that paper. □

The cyclic CD approach would seem to have an intuitive advantage over the full gradient steepest-descent method, if we compare a single cycle of cyclic CD to one step of the steepest-descent method. Cyclic CD is making use of the most current gradient information whenever it takes a step along a coordinate direction, whereas the steepest-descent method evaluates the moves along all n coordinates at the same value of x. This advantage is not reflected in the worst-case analysis of Theorem 6.3, which suggests slower convergence than the full gradient steepest-descent method, even when we assume that the cost per iteration differs by $O(n)$ between the two approaches (see details in what follows). Indeed, the proof of Beck and Tetruashvili (2013) treats the cyclic CD method as a kind of perturbed steepest-descent method, bounding the change in objective value over one cycle in terms of the gradient at the start of the cycle.

The bounds (6.20) and (6.21) are generally worse than the corresponding bounds (6.7) and (6.8) obtained for the randomized algorithm, as we explain in a moment. Computational comparisons between randomized and cyclic methods show similar performance on many problems, but as a comparison of the bounds suggests, cyclic methods perform worse (sometimes much worse) when the ratio $L/L_{\max}$ significantly exceeds its lower bound of 1. We note also that the bounds (6.20) and (6.21) are deterministic, whereas (6.7) and (6.8) are bounds on *expected* suboptimality.

We illustrate the results of Theorem 6.3 with three possible choices for α. Setting α to its upper bound of $1/L_{\max}$, we have for (6.20) that

$$f(x^k) - f^* \le \frac{4nL_{\max}(1 + nL^2/L_{\max}^2)R_0^2}{k+8} \approx \frac{4n^2L^2R_0^2}{kL_{\max}}.$$

The numerator here is worse than the corresponding result (6.7) by a factor of approximately $2nL^2/L_{\max}^2 \in [2n, 2n^3]$, suggesting better performance for the randomized method, with a larger advantage on problems for which $L_{\max} \ll L$. If we set $\alpha = 1/L$ (a valid choice, since $L \ge L_{\max}$), (6.20) becomes

$$\frac{4n(n+1)LR_0^2}{k+8} \approx \frac{4n^2LR_0^2}{k},$$

which is worse by a factor of approximately $2n^2$ than the bound (6.16) for the full-step gradient descent approach. For $\alpha = 1/(\sqrt{n}L)$, we obtain

$$\frac{8n^{3/2}LR_0^2}{k+8},$$

which still trails (6.16) by a factor of $4n^{3/2}$. If we take into account the factor-of-n difference in cost between iterations of CD and full gradient methods for problems of interest, these differences shrink to factors of n and $n^{1/2}$, respectively.

A different analysis (due to Sun and Hong, 2015, section 3) for weakly convex f yields a $1/k$ sublinear rate, like (6.20), but the constant has different dependences on the various Lipschitz constants. The constant can be significantly smaller in some cases, when $L/L_{\max}$ near its upper bound of n, but larger in other cases.

6.2.4 Random Permutations CD: Sampling without Replacement

The *random-permutations* variant of CD is a kind of hybrid of the randomized and cyclic approaches. As in the cyclic approach, the computation are divided into epochs of n iterations each, where within each epoch, every coordinate is updated exactly once. Unlike the cyclic approach, however, the coordinates are shuffled at the start of each epoch. (Equivalently, we can think of the iterations within each epoch as sampling the coordinates from the set $\{1,2,\dots,n\}$ *without replacement*.)

The convergence properties proved for cyclic CD in Theorem 6.3 continue to hold for random-permutations CD; the proofs in Beck and Tetruashvili (2013) need no modification. Curiously, however, computational experience shows that the random-permutations variant avoids the poor behavior of the purely cyclic variant in cases for which the ratio $L/L_{\max}$ is large. In all cases, performance is quite similar to that of "sampling with replacement" – the randomized CD approach of Section 6.2.2. This behavior is explained analytically in some special cases, a particular strongly convex quadratic in Lee and Wright (2018) and for a more general class of strongly convex quadratics in Wright and Lee (2020). Even in these special cases, the analysis of random-permutations CD is much more complex than for either randomized CD or cyclic CD.

6.3 Block-Coordinate Descent

All methods described in this chapter can be extended to the case in which the coordinates are partitioned into *blocks*, each of which contains one or more components. After possible rearrangement of the components of x, we can partition it as

$$x = (x_{(1)}, x_{(2)}, \dots, x_{(p)}),$$

where $x_{(i)} \in \mathbb{R}^{n_i}$, $i = 1,2,\dots,p$ and $\sum_{i=1}^{p} n_i = n$. We use U_i to denote those columns of the $n \times n$ identity matrix that correspond to the components in $x_{(i)}$. Generalizing (6.2), step k of the block-coordinate descent method can thus be defined as follows:

$$x^{k+1} \leftarrow x^k - \alpha_k U_{i_k} \nabla_{i_k} f(x^k),$$

for some $\alpha_k > 0$, where $\nabla_i f(x)$ is the vector of partial derivatives of f with respect to the components in $x_{(i)}$. Definitions of the componentwise Lipschitz constants (6.4) can be extended trivially to blocks, as follows:

$$\|\nabla_i f(x + U_i v_i) - \nabla_i f(x)\| \le L_i \|v_i\|, \quad \text{any } v_i \in \mathbb{R}^{n_i}, \text{ all } i = 1,2,\dots,p. \tag{6.22}$$

Some algorithms also make use of moduli of convexity m_i on the blocks, where m_i satisfies $m_i I \preceq U_i^T \nabla^2 f(x) U_i \in \mathbb{R}^{n_i \times n_i}$, for all x in the domain of interest.

The block-coordinate structure allows many algorithms to be extended to the case when *block-separable regularizers* are present. These problems have objectives of the form

$$f(x) + \sum_{i=1}^{p} \Omega_i(x_{(i)}), \tag{6.23}$$

where each Ω_i is convex and often nonsmooth.

Generalization of the analysis of randomized CD and cyclic CD methods is straightforward; in fact, several of our sources for this chapter describe the results in the block-coordinate framework rather than the single-coordinate setting that we adopted here (see, for example, Beck and Tetruashvili, 2013; Nesterov, 2012; Nesterov and Stich, 2017; Lu and Xiao, 2015).

Block-coordinate descent is a natural technique to apply in several applications. For example, in low-rank matrix completion, given observations of the (i,j) elements of a matrix $M \in \mathbb{R}^{p\times q}$, where $(i,j) \in \mathcal{O} \subset \{1,2,\dots,p\} \times \{1,2,\dots,q\}$, we seek matrices $U \in \mathbb{R}^{p\times r}$ and $V \in \mathbb{R}^{q\times r}$ (for some $t \le \min(p,q)$) to minimize the objective

$$f(U,V) := \sum_{(i,j)\in\mathcal{O}} \left([UV^T - M]_{i,j}\right)^2.$$

A natural approach is to define two blocks of variables – U and V – and minimize successively with each of these blocks. Since $f(U,V)$ is a least-squares problem in U for fixed V, and a least-squares problem in V for fixed U, standard methods are available for minimizing over the blocks. Tensor completion problems can be handled in a similar way; once again, the subproblems are least-squares problems.

In *nonnegative* matrix factorization, we further constrain U and V to have only nonnegative elements. The objective for the problem can be stated in the form (6.23) – that is,

$$\sum_{(i,j)\in\mathcal{O}} \left([UV^T - M]_{i,j}\right)^2 + I^+_{p,r}(U) + I^+_{q,r}(V),$$

where I^+ are indicator functions that have the value 0 if all elements of the matrix are nonnegative and ∞ otherwise. The subproblems in this formulation are bound-constrained least-squares problems, which can be solved with projected gradient or active-set methods.

Notes and References

A notable early paper on block-coordinate descent with block-separable regularization is by Tseng and Yun (2010), who proved convergence and complexity rates for several settings (including nonconvex f) and different variants. This paper assumes that the block of variables chosen for updating at each step satisfies a generalized Gauss–Southwell condition, which ensures that the improvement in f by considering this block is a nontrivial fraction of the improvement available from a full gradient step. Some extensions of this paper are considered by Wright (2012), together with a discussion of several applications and local convergence results.

The proof of Theorem 6.1 is a simplified version of the analysis in Nesterov (2012, section 2). The proof of Theorem 6.2 is from Lu and Xiao (2015, theorem 1). Another analysis (extendable to problems with separable regularizers) is given by Richtarik and Takac (2014).

Variants of the randomized CD approach that make use of Nesterov acceleration were proposed first by Nesterov (2012), with a version that can be implemented efficiently in some applications proposed later in Lee and Sidford (2013). A more generally applicable version is described by Nesterov and Stich (2017). When the sampling probability for component i is chosen to be $L_i^{1/2}/(\sum_{j=1}^n L_j^{1/2})$ (where the L_i are the componentwise Lipschitz constants defined in (6.4)), Nesterov and Stich (2017, theorem 1) proves the bound

$$\mathbb{E}(f(x^k) - f(x^*)) \le \frac{2R_0^2 \left(\sum_{i=1}^n L_i^{1/2}\right)^2}{k^2}.$$

(Note that the $1/k$ rate of Theorems 6.1 and 6.2 has been replaced by the $1/k^2$ rate that is typical of accelerated methods.)

Analysis of the cyclic method in Section 6.2.3 is from Beck and Tetruashvili (2013). A later work (Li et al., 2018) uses techniques similar to those of Sun and Hong (2015) to analyze a version of cyclic CD with a particular choice of step sizes, related to the componentwise Lipschitz constants, for the strongly convex case. Some improvements to the complexity results of Beck and Tetruashvili (2013) are obtained for these cases. (The different setting and assumptions make it difficult to compare the results directly, but the bound on the number of iterates required to attain a specified accuracy in the objective function is approximately a factor of $L/L_{\max}$ better in Li et al., 2018.) The techniques of Li et al. (2018) apply to the case in which separable nonsmooth regularization terms also appear in the objective; we consider the extension of CD methods to such problems in Section 9.4.

When the function f satisfies the Polyak–Łojasiewicz (PL) condition of Section 3.8, Karimi et al. (2016) shows linear convergence rates for algorithms of the form (6.2) and its extensions to problems with separable regularization terms. Here, as before, the PL condition yields results similar to those obtained for strongly convex functions. Chouzenoux et al. (2016)

consider block-coordinate descent for the separable regularized case, using variable metrics to modify the gradient at each iteration, and proves global convergence as well as local convergence rates under the Kurdyka–Łojasiewicz (KL) condition (which is also described in Section 3.8).

The justification for using CD methods as opposed to full gradient methods is perhaps seen best in asynchronous implementations on parallel computers. Multiple cores can, of course, share the workload of evaluating a full gradient, but there is inevitably a synchronization point – the computation must wait for all cores to complete their share of the work before it can proceed with computing and taking the step. Asynchronous implementations of CD methods are easy to design, especially for multicore, shared-memory computers in which all cores have access to a shared version of the variable x (and possibly other quantities involved in the evaluation of gradient information). Strong results about the convergence of asynchronous algorithms under weak assumptions were obtained by Bertsekas and Tsitsiklis (1989, section 7.5). More recently, several papers (Liu et al., 2015; Liu and Wright, 2015) showed that convergence rates of the serial CD methods are largely inherited by multicore implementations provided that the number of cores is not too large. Other parallel implementations have also been devised, analyzed, and implemented in Richtarik and Takac (2016b), Fercoq and Richtarik (2015), and Richtarik and Takac (2016a). Parallel CD remains an active area of research.

We note that the analysis in the papers cited in this chapter defines the constant R_0 differently from in Assumption 1, to be $\max_{x \in \mathcal{L}^0} \max_{x^* \in \mathcal{S}} \|x - x^*\|$ rather than $\max_{x \in \mathcal{L}^0} \min_{x^* \in \mathcal{S}} \|x - x^*\|$. This alternative definition results of course in a larger value, which has the disadvantage of being infinite when the solution set is unbounded. A careful look at the analysis of these papers shows that the definition that we use here suffices.

Exercises

1. In the ERM example of Section 6.1, assume that the objective function f is known at the current point x, along with the quantity $g = Ax$. Show that the cost of computing $f(x + \gamma_i e_i)$ for some $i = 1, 2, \dots, n$ is $O(|A_{\cdot i}|)$ (the number of elements in column i of A) – the same order as the cost of updating the gradient ∇f. Show that a similar observation holds for the graph example in Section 6.1.
2. Consider the convex quadratic $f(x) = \frac{1}{2} x^T A x$ with $A = ee^T$, where $e = (1, 1, \dots, 1)^T$, for which $L = n$ and $L_{\max} = L_i = 1$ for $i = 1, 2, \dots, n$. Show that any variant of CD with $\alpha = 1/L_{\max}$ or $\alpha = 1/L_i$

converges in one iteration. Show that the steepest-descent method (with either exact line search or steplength $\alpha = 1/L$) also converges in one step.

3. Implement the following variants of coordinate descent:
   - Randomized CD (the method of Section 6.2.2) with exact line search and with constant steplength $1/L_{\max}$
   - Cyclic CD (Section 6.2.3) with exact line search and with constant steplengths $1/L_{\max}$, $1/L$, and $1/(\sqrt{n}L)$
   - Random-permutations CD (Section 6.2.4) with exact steps and and with constant steplength $1/L_{\max}$.

   Compare the performance of these methods on convex quadratic problems $f(x) = \frac{1}{2}x^T Ax$, where A is an $n \times n$ positive semidefinite matrix constructed randomly in the manner described in what follows. (Note that $x^* = 0$ with $f(x^*) = 0$.) Terminate when $f(x) \le 10^{-6} f(x^0)$. Use a random starting point x^0 whose components are uniformly distributed in $[0, 1]$. Compute and print the values of L and $L_{\max}$ for each instance.

   Test your code on the following matrices A.

   (i) $A = Q^T DQ^T$, where Q is random orthogonal and D is a positive diagonal matrix whose here each diagonal D_{ii} has the form $10^{-\zeta_i}$, where each ζ_i is drawn uniformly i.i.d. from $[0, 1]$.

   (ii) The same as in (i), but with each ζ_i drawn uniformly i.i.d. from $[0, 2]$.

   (iii) Generate the matrix A as in (i), then replace it by $A + 10ee^T$, where $e = (1, 1, \ldots, 1)^T$.

   Discuss the relative performance of the methods on these different problems. How is your computational experience consistent (or inconsistent) with the convergence expressions obtained in Theorems 6.1 and 6.3?

4. Compare the linear convergence bounds (6.8) and (6.21) for randomized CD and cyclic CD, for various choices of steplength in the cyclic method, including $\alpha = 1/L_{\max}$, $\alpha = 1/L$, and $\alpha = 1/(\sqrt{n}L)$. (In making these comparisons, note that, for small ϵ, we have $(1 - \epsilon)^{1/n} \approx 1 - \epsilon/n$.) Which of these choices of fixed steplength α in the cyclic method is optimal, in the sense of approximately minimizing the factor on the right-hand side of (6.21)?