DSC 211 Introduction to Optimization Winter 2024 Instructor: Jun-Kun Wang
Scribe: Merlin Chang, Marialena Sfyraki January 25, 2024

Lecture 6: Coordinate Descent (CD)

1 (Randomized) Coordinate Descent

1.1 Algorithm

Consider the general unconstrained optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}).$$

Recall the definition of the gradient:

Definition 1. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a differentiable function. Then, the function $\nabla f : \mathbb{R}^d \to \mathbb{R}^d$ defined by*

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}[1]} \\ \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}[2]} \\ \vdots \\ \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}[d]} \end{bmatrix} \in \mathbb{R}^d$$

is called the gradient of f at $\mathbf{x}$.

Based on the previously described concept of the gradient, we are now able to present a detailed formulation of the (Randomized) Coordinate Descent algorithm as follows:

Algorithm 1 (Randomized) Coordinate Descent

1: **for** $k = 1, 2, \ldots$ **do**
2: Randomly pick a coordinate $i_k \in [d]$.
3: $\mathbf{x}_{k+1}[i_k] = \mathbf{x}_k[i_k] - \eta \nabla f(\mathbf{x}_k)[i_k]$.
4: **end for**

Remark: Observe that only the i_k-th element is updated at each timestep.

1.2 Advantags of Coordinate Descent (CD)

The examples below demonstrate the advantages of the introduced Coordinate Descent algorithm in comparison to the earlier discussed Gradient Descent.

Case I: Let $f(x) = \log\Big(1 + exp\left(-yz^\top x\right)\Big)$. Then, the gradient of f can be computed as

$$\nabla f(x) = \frac{exp\left(-yz^\top x\right)}{1 + exp\left(-yz^\top x\right)}\left(-yz\right) \in \mathbb{R}^d.$$

Thus, the number of computations/flops for the GD algorithm are

$$\#\text{computations/flops} = \underbrace{d}_{\text{dot product}} + \underbrace{1}_{\text{addition}} + \underbrace{1}_{\text{multiplication}} + \underbrace{1}_{\text{division}} + \underbrace{d}_{\text{dot product}} = 2d + 3.$$

In computing, when we are concerned with the number of floating point operations computed per second, we use the short hand measure of flops. Although the true computation cycles needed for addition and multiplication are very different, we are omitting these details in our analysis.
Additionally, the i-th coordinate of the gradient is

$$\left[\nabla f(x)\right]_i = \frac{exp\left(-yz^\top x\right)}{1 + exp\left(-yz^\top x\right)}\left(-yz\left[i\right]\right) \in \mathbb{R}.$$

Thus, the number of computations/flops for the CD algorithm are

$$\#\text{computations/flops} = \underbrace{d}_{\text{dot product}} + \underbrace{1}_{\text{addition}} + \underbrace{1}_{\text{multiplication}} + \underbrace{1}_{\text{division}} + \underbrace{1}_{\text{addition}} = d + 4.$$

Therefore, we can conclude that

$$\frac{\text{Cost of CD}}{\text{Cost of GD}} \approx \frac{1}{2}$$

which is not a significant speedup.
Case II: Let $f(x) = \frac{1}{2}x^\top Ax - b^\top x$, where $A \in \mathbb{R}^{d\times d}$. The gradient of f can be computed as

$$\nabla f(x) = Ax - b.$$

Thus, the number of computations/flops for the GD algorithm are

$$\#computations/flops = d^2 + d.$$

Additionally, the i-th coordinate of the gradient is

$$\left[\nabla f(x)\right]_i = \left[Ax - b\right]_i = \sum_{j=1}^{d} A_{ij}x_j + b_i.$$

Thus, the number of computations/flops for the CD algorithm are

$$\#computations/flops = d + 1.$$

In this case, we can conclude that

$$\frac{\text{Cost of CD}}{\text{Cost of GD}} = \frac{1}{d}.$$

Remark. The extent of superiority of CD over GD varies with the objective function.

1.3 Coordinate-wise Smooth

Definition 2. *We say a function $f(\cdot) : \mathbb{R}^d \to \mathbb{R}$ is coordinate-wise smooth if for all $i \in [d]$ and $x \in \mathbb{R}^d$, the function*

$$g_{i,x}(v) := f(x + v\,\mathbf{e}_i) : \mathbb{R} \to \mathbb{R}, \mathbf{e}_i \text{ is the standard basis}$$

satisfies

$$g_{i,x}(v) \leq g_{i,x}(0) + \nabla g_{i,x}(0)v + \frac{L_i}{2}v^2,$$

for any $v \in \mathbb{R}$ and a finite constant $L_i > 0$.

Remark. Observe that

$$\nabla g_{i,x}(v) = \langle \nabla f\left(x + v\mathbf{e}_i\right), \mathbf{e}_i \rangle = \left[\nabla f\left(x + v\mathbf{e}_i\right)\right]_i$$

and if f is twice continuously differentiable, then

$$\nabla^2 g_{i,x}(v) = \langle \nabla^2 f\left(x + v\mathbf{e}_i\right)\mathbf{e}_i, \mathbf{e}_i \rangle = \left[\nabla^2 f\left(x + v\mathbf{e}_i\right)\right]_{i,i}.$$

Recall the Fundamental Theorem of Calculus:

Theorem 1 (Fundamental Theorem of Calculus)**.** *Let $h : \mathbb{R} \to \mathbb{R}$ be a continuously differentiable function. Then,*

$$h(b) - h(a) = \int_a^b \dot{h}(\alpha)\, d\alpha.$$

Remark. An equivalent definition of coordinate-wise smoothness when the function $f(\cdot)$ is convex is

$$|\nabla g_{i,x}(v) - \nabla g_{i,x}(0)| \leq L_i\,|v|\,, \forall i \in [d], \forall x \in \mathbb{R}^d,$$

that is the gradient is Lipschitz. Furthermore, if the function $f(\cdot)$ is twice continuously differentiable and we have

$$\left[\nabla^2 f\left(x+\alpha \mathbf{e}_i\right)\right]_{i,i} \leq L_i \,, \forall i \in [d], \forall x \in \mathbb{R}^d, \forall \alpha \in [0, v] \tag{1}$$

then, by the Fundamental Theorem of Calculus

$$\left|\nabla g_{i,x}(v) - \nabla g_{i,x}(0)\right| = \left|\int_0^v \nabla^2 g_{i,x}(\alpha) d\alpha\right| = \left|\int_0^v \left[\nabla^2 f\left(x+\alpha \mathbf{e}_i\right)\right]_{i,i} d\alpha\right| \leq L_i \left|v\right|, \tag{2}$$

$\forall i \in [d], \forall x \in \mathbb{R}^d$.

Hence, we can relate the smoothness constant of $f(\cdot)$ and the coordinate-wise smoothness of L_i as follows:

$$L = \lambda_{max}\left(\nabla^2 f\left(x\right)\right) \leq tr\left(\nabla^2 f\left(x\right)\right) \leq \sum_{i=1}^{d} L_i. \tag{3}$$

1.4 Iteration Complexity of Randomized Coordinate Descent

Theorem 2. *Suppose $f(\cdot) : \mathbb{R}^d \to \mathbb{R}$ be μ-strongly convex and satisfies coordinate-wise smooth with the constant L_i for each $i \in [d]$. Denote $S := \sum_i L_i$. Pick $i_k = i$ with $Pr[i_k = i] = \frac{L_i}{\sum_{j=1} L_j} = \frac{L_i}{S}$. Then,*

$$\mathbb{E}[f(x_{K+1})] - f_* \leq \left(1 - \frac{\mu}{S}\right)^K (f(x_1) - f_*).$$

Remark. Using the above theorem, we can see that in order to get an ϵ-expected optimality gap for a μ-strongly convex and coordinate-wise smooth function f using the CD algorithm, the number of iterations K should be

$$K = \frac{S}{\mu} \log\left(\frac{f\left(x_1\right) - f_*}{\epsilon}\right).$$

We have proven previously that in order to get an ϵ optimality gap for for a μ-strongly convex and L-smooth function f using the GD algorithm, the number of iterations K should be

$$K = \frac{L}{\mu} \log\left(\frac{f\left(x_1\right) - f_*}{\epsilon}\right).$$

Question. How can we compare $S = \sum_i L_i$ and L?
From the coordinate-wise smoothness (1) and (2), we can estimate L_i as

$$L_i \approx \left[\nabla^2 f\left(x\right)\right]_{i,i}.$$

Then,

$$S = \sum_{i=1}^{d} L_i \approx \sum_{i=1}^{d} \left[\nabla^2 f(x)\right]_{i,i} = tr\left(\nabla^2 f(x)\right).$$

Additionally, by L-smoothness we know that

$$L = \lambda_{max}\left(\nabla^2 f(x)\right).$$

Thus,

$$\frac{S}{L} \approx \frac{tr\left(\nabla^2 f(x)\right)}{\lambda_{max}\left(\nabla^2 f(x)\right)}.$$

This is roughly the ratio between the runtime of gradient descent and coordinate descent, since

$$\frac{\#\text{ iterations of CD}}{\#\text{ iterations of GD}} \approx \frac{O(\frac{S}{\mu}\log(\frac{1}{\epsilon}))}{O(\frac{L}{\mu}\log(\frac{1}{\epsilon}))} \approx \frac{S}{L} \approx \frac{tr\left(\nabla^2 f(x)\right)}{\lambda_{max}\left(\nabla^2 f(x)\right)}.$$

If we take the cost per iteration for both algorithms into account we have

$$\frac{\text{CD running time}}{\text{GD running time}} = \frac{\#\text{ iterations of CD}}{\#\text{ iterations of GD}} \times \frac{\text{cost per k of CD}}{\text{cost per k of GD}} \approx \frac{tr\left(\nabla^2 f(x)\right)}{\lambda_{max}\left(\nabla^2 f(x)\right)} \times \frac{\text{cost per k of CD}}{\text{cost per k of GD}}.$$

Remark. Observe that when the spectrum (distribution of eigenvalues) of the Hessian is skewed so that $\mathrm{tr}(\nabla^2 f(x))$ is not very much larger than $\lambda_{\max}(\nabla^2 f(x))$, then CD might have an advanatage over GD.

1.5 Progress of a coordinate descent step

Lemma 1. *If we set $\eta_k = \frac{1}{L_{i_k}}$, then*

$$f(x_{k+1}) \leq f(x_k) - \frac{1}{2L_{i_k}}(\nabla f(x_k)[i_k])^2.$$

Proof. Recall that we have

$$g_{i,x}(v) := f(x + v\,\mathbf{e}_i) : \mathbb{R} \to \mathbb{R},$$

and

$$g_{i,x}(v) \leq g_{i,x}(0) + \nabla g_{i,x}(0)v + \frac{L_i}{2}v^2,$$

where

$$\nabla g_{i,x}(v) = \langle \nabla f(x + v\mathbf{e}_i), \mathbf{e}_i \rangle = \left[\nabla f(x + v\mathbf{e}_i)\right]_i.$$

Let

- $v \leftarrow -\eta_k \nabla f(x_k)[i_k]$ and $\eta_k = \frac{1}{L_{i_k}}$.
- $x \leftarrow x_k$.
- $i \leftarrow i_k$.

Since

$$x_k + v\mathbf{e}_i = x_k - \eta_k \nabla f(x_k)[i_k]\mathbf{e}_i = x_{k+1}$$

Then, we have

$$\underbrace{g_{i_k,x_k}\left(\eta_k \nabla f(x_k)[i_k]\right)}_{=f(x_{k+1})} \leq \underbrace{g_{i_k,x_k}(0)}_{=f(x_k)} - \underbrace{\nabla g_{i_k,x_k}(0)}_{=\nabla f(x_k)[i_k]} \eta_k \nabla f(x_k)[i_k] + \frac{L_{i_k}}{2}\eta_k^2 \left(\nabla f(x_k)[i_k]\right)^2 \quad (4)$$

$$= f(x_k) - \frac{1}{L_{i_k}}(\nabla f(x_k)[i_k])^2 + \frac{L_{i_k}}{2}\frac{1}{{L_{i_k}}^2}(\nabla f(x_k)[i_k])^2 \quad (5)$$

$$= f(x_k) - \frac{1}{2L_{i_k}}(\nabla f(x_k)[i_k])^2. \quad (6)$$

□

1.6 Progress at iteration k in expectation

Lemma 2. *Suppose f satisfies the conditions of Theorem 2. At iteration k, randomly pick $i_k \in [d]$ with probability $\Pr[i_k = i] = p_i$. Then,*

$$\mathbb{E}_{i_k}[f(x_{k+1})] \leq f(x_k) - \sum_{i=1}^{d} \frac{p_i}{2L_i}(\nabla f(x_k)[i])^2.$$

Proof. By Lemma 1, we have

$$f(x_{k+1}) \leq f(x_k) - \frac{1}{2L_i}(\nabla f(x_k)[i])^2.$$

Then the expectation of the $f(x_{k+1})$ is

$$\begin{aligned}
\mathbb{E}\left[f(x_{k+1})\right] &= \sum_{i=1}^{d} p_i f(x_{k+1}) \\
&\leq \sum_{i=1}^{d} \left(p_i f(x_k) - \frac{p_i}{2L_i}(\nabla f(x_k)[i])^2\right) \qquad \text{, by Lemma 1} \\
&= f(x_k) - \sum_{i=1}^{d} \frac{p_i}{2L_i}(\nabla f(x_k)[i])^2.
\end{aligned}$$

□

1.7 Proof for Theorem (2)

Recall Theorem 2: *Suppose $f(\cdot) : \mathbb{R}^d \to \mathbb{R}$ be μ-strongly convex and satisfies coordinate-wise smooth with the constant L_i for each $i \in [d]$. Denote $S := \sum_i L_i$. Pick $i_k = i$ with $Pr[i_k = i] = \frac{L_i}{\sum_{j=1} L_j} = \frac{L_i}{S}$. Then,*

$$\mathbb{E}[f(x_{K+1})] - f_* \leq \left(1 - \frac{\mu}{S}\right)^K (f(x_1) - f_*).$$

Proof. By Lemma 2, we know that

$$\begin{aligned}
\mathbb{E}_{i_k}[f(x_{k+1})] &\leq f(x_k) - \sum_{i=1}^{d} \frac{p_i}{2L_i} (\nabla f(x_k)[i])^2 \\
&= f(x_k) - \frac{1}{2S} \sum_{i=1}^{d} (\nabla f(x_k)[i])^2 && \text{, since } p_i = \frac{L_i}{S} \\
&= f(x_k) - \frac{1}{2S} \|\nabla f(x_k)\|_2^2 && \text{, by 2-norm definition} \\
&\leq f(x_k) - \frac{\mu}{S} \left(f(x_k) - f_*\right) && \text{, by } \mu\text{-gradient dominance}
\end{aligned}$$

By subtracting f_* on both sides, we get

$$\begin{aligned}
&\mathbb{E}_{i_k}[f(x_{k+1})] - f_* \leq f(x_k) - \frac{\mu}{S} \left(f(x_k) - f_*\right) - f_* \\
&\Leftrightarrow \mathbb{E}_{i_k} \left[f(x_{k+1}) - f_*\right] \leq \left(1 - \frac{\mu}{S}\right) \left(f(x_k) - f_*\right).
\end{aligned}$$

Denote $\delta_{k+1} := f(x_{k+1}) - f_*$. Then, we have

$$\begin{aligned}
\mathbb{E}_{i_1,\ldots,i_k}[\delta_{k+1}] &= \mathbb{E}_{i_1,\ldots,i_{k-1}} \left[\mathbb{E}_{i_k}[\delta_{k+1} | i_1 \ldots, i_{k-1}] \right] && \text{, by the law of total expectation} \\
&\leq \mathbb{E}_{i_1,\ldots,i_{k-1}} \left[\left(1 - \frac{\mu}{S}\right) \delta_k \right] && \text{, by previously shown inequality} \\
&= \left(1 - \frac{\mu}{S}\right) \mathbb{E}_{i_1,\ldots,i_{k-1}}[\delta_k] && \text{, since } x_k \text{ is determined by } i_1, \ldots, i_{k-1} \\
&\leq \left(1 - \frac{\mu}{S}\right) \left(1 - \frac{\mu}{S}\right) \mathbb{E}_{i_1,\ldots,i_{k-2}}[\delta_{k-1}] \\
&\leq \left(1 - \frac{\mu}{S}\right)^k \delta_1 && \text{, since } \delta_1 \text{ is deterministic} \\
&= \left(1 - \frac{\mu}{S}\right)^k (f(x_1) - f_*)
\end{aligned}$$

□

2 Stochastic Optimization

2.1 Algorithm

Now we will look at Stochastic Gradient Descent (SGD). Our objective is to

$$\min_x F(x)$$

where $F(x) := \mathbb{E}_z[f(x; z)]$.

Algorithm 2 Stochastic Gradient Descent

1: **for** $k = 1, 2, \ldots$ **do**
2: Compute a stochastic gradient g_k that satisfies $\mathbb{E}_z[g_k] = \nabla F(x_k)$
3: $x_{k+1} = x_k - \eta g_k$.
4: **end for**

Remark. Randomized Coordinate Descent is a particular instance of SGD. Recall the update step of Randomized Coordinate Descent

$$x_{k+1}[i_k] = x_k[i_k] - \eta_k \nabla f\left(x_k\right)[i_k].$$

The update step can be written equivalently as

$$x_{k+1} = x_k - \eta_k \nabla \left[f\left(x_k\right)\right]_{i_k} \mathbf{e}_{i_k}.$$

Then, we can observe that

$$g_k = \nabla \left[f\left(x_k\right)\right]_{i_k} \mathbf{e}_{i_k}.$$

Note that

$$\mathbb{E}[g_k] = \mathbb{E}\left[\nabla \left[f\left(x_k\right)\right]_{i_k} \mathbf{e}_{i_k}\right] = \sum_{i=1}^{d} p_i \nabla \left[f\left(x_k\right)\right]_i \mathbf{e}_i = \frac{1}{d} \nabla f(x_k) \in \mathbb{R}^d,$$

so up to a scaling d (which can be absorbed in the step size η), it can be viewed as an unbiased estimate of the full gradient $\nabla f(x_k)$.

Remark. A stochastic gradient g_k for a function F at point x_k is the gradient, whose expectation is the full gradient, $\mathbb{E}_z[g_k] = \nabla F(x_k)$.

One way to interpret $F(x)$ is to think in terms of a linear regression model. For example, in the case of linear regression, the model's loss function, f is in the term

$$(y - z^\top x)^2$$

where (y, z) is a particular labeled instance in the training set. What we ultimately want to minimize if the following:

$$\min_x \mathbb{E}_z \left[(y - z^\top x)^2 \right]$$

which is the same as

$$\min_x \mathbb{E}_z \left[(f(x; z) \right]$$

This is saying that we want to minimize the expected loss of our model when sampling z from an underlying distribution. In practice, we do not have access to the distribution of z, namely the distribution of labeled data, instead only a sampled subset. We can in turn treat the sampled set(training set) as the distribution and minimize the empirical loss, which is defined as:

$$\min_x \frac{1}{n} \sum_{i=1}^{n} (y_i - z_i^\top x)^2$$

This finite-sum problem is refererd to as empirical risk minimization in machine learning literature.

Bibliographic notes

Part of the materials in this lecture is based on Section 6.3 of [1].

References

[1] Aaron Sidford Optimization Algorithms 2023