ECE 901: Large-scale Machine Learning and Optimization **Spring 2018**

Lecture 7 — 02/15

Lecturer: Dimitris Papailiopoulos *Scribe: Boyu Zhang, Haoran Zhu*

Note: These lecture notes are still rough, and have only have been mildly proofread.

This is the danger environment.

7.1 Review

Coordinate descent algorithms for optimization have a history that dates to the foundation of the discipline. They are iterative methods in which each iterate is obtained by fixing most components of the variable vector x at their values from the current iteration, and approximately minimizing the objective with respect to the remaining components. Various applications (including several in computational statistics and machine learning) have yielded problems for which CD approaches are competitive in performance with more reputable alternatives, some of the properties of these problems, such as the low cost of calculating one component of the gradient, lend themselves well to efficient implementations of coordinate descent. In this scribing we're going to breifly talk about the most widely used variant of the CD family: randomized coordinate descent.

7.2 Radomized Coordinate Descent

For $f : \mathbb{R}^d \to \mathbb{R}$, and we want to minimize $f(x)$, then the RCD algorithm looks like the following:
$x^{k+1} = x^k - \gamma \nabla_{s_k} f(x^k) \cdot e_{s_k}$, here s_k has several selection options:
- $s_k \sim unit\{1, 2, \ldots, d\}$(uniformly at random)
- $p(s_k = i) = p_i$ (importance sampling)
- Greedy selection.

Next we're going to make several assumptions and correspondingly we give the lemma under those assumptions.

7.2.1 $\lambda-$strongly convex

Lemma 7.1. *Suppose f is $\lambda-$strongly convex, then $\forall x, f(x) - f^* \leq \frac{1}{2\lambda}\|\nabla f(x)\|^2$.*

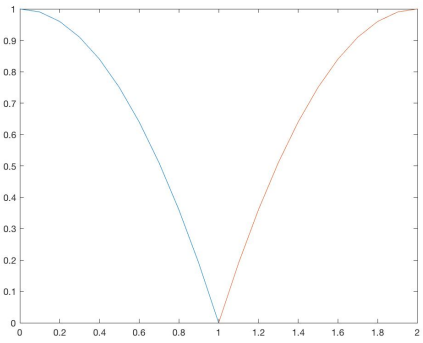
Proof: From the $\lambda-$strongly convex assumption we get: $f(y) \geq f(x) + < \nabla f(x), y - x > + \frac{\lambda}{2}\|y - x\|^2, \forall y, x$. Then put $y = x - \gamma \nabla f(x)$ into above, and we can get: $f(y) \geq$

$f(x) - \gamma\|\nabla f(x)\|^2 + \frac{\lambda}{2}\|\nabla f(x)\|^2$, minimize the right hand side over stepsize γ, we can have: $f(y) \geq f(x) - \frac{1}{2\lambda}\|\nabla f(x)\|^2$. By setting $y = x^*$ we have $f(x) - f^* \leq \frac{1}{2\lambda}\|\nabla f(x)\|^2$. □

7.2.2 Polyak-Lojasiewicz condition

Definition 1. *We say a function f satisfies Polyak-Lojasiewicz condition, if:* $f(x) - f^* \leq \frac{1}{2\lambda}\|\nabla f(x)\|^2$.

Note: The reason for introducing this condition is, gradient descent method usually converges nicely under this condition.
Clearly from the last lemma we already known $\lambda-$strongly convex functions are also "PL" functions, but there are also many nonconvex cases, it can be seen as the non-convex generalization of $\lambda-$strongly convex.



examples: $f(x) = x^2 + 3sin^2 x$ is $\lambda-$PL; $f(x, y) = (xy-1)^2$ is $\lambda-$PL on bounded regions.

7.2.3 Component-wise $\beta-$smooth

Definition 2. *We say a function $f(x)$ is $\beta-$smooth for each coordinate, if for all i, there is a β_i, s.t.* $|\nabla_i f(x) - \nabla_i f(x + \alpha e_i)| \leq |\alpha|\beta_i$.

Note: This assumption is somewhat weaker than β-smoothness for whole function, which is $\|\nabla f(x) - \nabla f(y)\| \leqslant \beta|x - y|$.

7.3 Convergence rate of Randomized Coordinate Descent

Lemma 7.2. *If all the above assumptions hold, then* $f(x + \alpha e_i) \leqslant f(x) + \alpha\nabla_i f(x) + \frac{\beta_i\alpha^2}{2}$.

Proof: From Taylor expansion we have:

$$f(y) = f(x) + < \nabla f(x), y - x > +(y - x)^T\nabla^2 f(z)(y - x) \tag{7.1}$$

Where z lies on the line segment between x and y. Plug $y = x + \alpha e_i$ into equation 7.1 we have:

$$\begin{aligned} f(x+\alpha e_i) &= f(x) + \alpha \nabla_i f(x) + \frac{(\alpha e_i)^T \nabla^2 f(z) \alpha e_i}{2} && (7.2)\\ &= f(x) + \alpha \nabla_i f(x) + \frac{\alpha^2 (\nabla^2 f(z))_{i,i}}{2} && (7.3)\\ &\leqslant f(x) + \alpha \nabla_i f(x) + \frac{\alpha^2 \beta_i}{2} && (7.4) \end{aligned}$$

□

Thus, we showed:

- $f(x) - f^* \leqslant \frac{\|\nabla f(x)\|^2}{2\lambda}$
- $f(x + \alpha e_i) \leqslant f(x) + \alpha \nabla_i f(x) + \frac{\alpha^2 \beta_i}{2}$

In Randomized Coordinate Descent we have:

$$x_{k+1} = x_k - \alpha \nabla_{s_k} f(x_k) e_{s_k} \tag{7.5}$$

Let $\beta = \max_i \beta_i$ and $\alpha = \frac{1}{\beta}$. Then we have:

$$\begin{aligned} f(x_{k+1}) &= f(x_k - \frac{\nabla_{s_k} f(x_k) e_{s_k}}{\beta}) && (7.6)\\ &\leqslant f(x_k) - \frac{\nabla_{s_k} f(x_k)^2}{\beta} + \frac{\beta_i \nabla_{s_k} f(x_k)^2}{2\beta^2} && (7.7)\\ &\leqslant f(x_k) - \frac{\nabla_{s_k} f(x_k)^2}{2\beta} && (7.8) \end{aligned}$$

Take expection at both side regarding to s_k,

$$\begin{aligned} \mathop{\mathbb{E}}_{s_k}[f(x_{k+1})] &\leqslant f(x_k) - \frac{\mathbb{E}[\nabla_{s_k} f(x_k)^2]}{2\beta} && (7.9)\\ &= f(x_k) - \frac{\sum_{i=1}^d \frac{1}{d} \nabla_i f(x_k)^2}{2\beta} && (7.10)\\ &= f(x_k) - \frac{\|\nabla f(x_k)\|^2}{2\beta d} && (7.11)\\ &\leqslant f(x_k) + (-\frac{\lambda}{d\beta})(f(x_k) - f^*) && (7.12) \end{aligned}$$

Subtract f^* from both side of Equation 7.12, we have:

$$\mathbb{E}[f(x_{k+1}) - f^*] \leqslant (1 - \frac{\lambda}{d\beta})(f(x_k) - f^*) \tag{7.13}$$

Recursively apply Equation 7.13, finally we have:

$$\mathbb{E}[f(x_{k+1}) - f^*] \leqslant (1 - \frac{\lambda}{d\beta})^{k+1}(f(x_0) - f^*) \tag{7.14}$$

Thus, in order to reach ϵ accuracy, the number of iteration needed in uniform sampling is:

$$T_\epsilon^{unif} = \frac{d\beta_{max}}{\lambda} log(\frac{f(x_0) - f^*}{\epsilon}) \tag{7.15}$$

Compare the convergence rate and the number of iteration needed to reach ϵ accuracy with those of Gradient Descent algorithm, we will find RCD with uniform sampling actually requires $O(d)$ more iterations. To improve this, we introduce importance sampling in the next subsection.

7.4 Variations of Randomized Coordinate Descent

- **Importance Sampling for RCD.**

  In standard RCD, the update on each coordinate is in the following form:

  $$x_{k+1} = x_k - \alpha \nabla_{s_k} f(x_k) e_{s_k} \tag{7.16}$$

  Where $s_k = i$ is uniformly picked.

  Here, instead of uniformly pick one coordinate and perform the update, we set $s_k = i$ with the probability equals to $\frac{\beta_i}{\sum_{i=1}^d \beta_i}$. That is, each coordinate is sampled proportionally to its "effect" on $f(x)$. We also set $\alpha_k = \frac{1}{\beta_{s_k}}$.

  By going through the almost same derivation as in the last subsection, we can prove that:

  $$\mathbb{E}[f(x_{k+1}) - f^*] \leqslant (1 - \frac{\lambda}{\sum_{i=1}^d \beta_i})^{k+1}(f(x_0) - f^*) \tag{7.17}$$

  $$T_\epsilon^{imp} = \frac{\sum_{i=1}^d \beta_i}{\lambda} log(\frac{f(x_0) - f^*}{\epsilon}) \tag{7.18}$$

  Remark: $\sum_{i=1}^d \beta_i$ could be significantly smaller than $d\beta_{max}$. Hence, importance sampling can be extremely helpful.

- **Gauss-Southwell.**

  The idea is to pick the $s_k = i$ such that $i = \arg\max_i |\nabla_i f(x_k)|$.

- **Block RCD.**

  Assume $\{1, 2, 3, ..., d\} = B_1 \bigcup B_2 \bigcup ... \bigcup B_r$. We pick a B_i randomly and update all the coordiantes in B_i.

- **(No name)**

  The idea is to turn the multi-variable optimization problem into one-variable optimization problem and solve it. We randomly pick an i and we fixed all $x_j, j \neq i$. Then we solve the problem $\min_{x_i} f(x)$. We repeat this process until reach ϵ accuracy.

7.5 Advantages over Coordinate Descent

- If the gradient of one coordinate ($\nabla_i f(x)$) is very expensive to compute, we don't want to compute it very often.
- Parallelizable.
- Works very well with "seperable" functions, ie. $f(x) = \sum_{i=1}^{d} g(x_i)$. For example, L1-regularization $\|X\|_1$.

  Example: Let the loss function be a linear least square function ($f(x) = \|Ax - b\|^2$), and we want to minimize this function ($\min_x \|Ax - b\|^2$).

  Thus, we have:

$$\nabla f(x) = A^T(Ax - b) \tag{7.19}$$
$$\nabla_i f(x) = A[i]^T(Ax - b) \tag{7.20}$$

  Where $A[i]$ is the ith column of A.

  According to β-smoothness of each coordiante, we have:

$$\|\nabla_i f(x + \alpha e_i) - \nabla_i f(x)\| \leqslant \alpha\beta_i \tag{7.21}$$
$$\Rightarrow A[i]^T A(\alpha e_i) \leqslant \alpha\beta_i \tag{7.22}$$
$$\Rightarrow \beta_i \geqslant A[i]^T A e_i \tag{7.23}$$
$$= A[i]^T A[i] \tag{7.24}$$
$$= \|A[i]\|^2 \tag{7.25}$$

  Let's say: $A = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & & & & \\ \vdots & & I_{d-1} & & \\ 1 & & & & \end{bmatrix}$

  Then

$$\|A[1]\|^2 = d \tag{7.26}$$
$$\sum \beta_i = \sum_{i=1}^{d} \|A[i]\|^2 \tag{7.27}$$
$$= \|A\|_F^2 \tag{7.28}$$
$$\approx d \tag{7.29}$$

Thus, the number of iteration required to reach ϵ accuracy for uniform sampling and importance sampling are:

$$T_\epsilon^{unif} = \frac{d\beta_{max}}{\lambda} log(\frac{f(x_0) - f^*}{\epsilon}) = \frac{d^2}{\lambda} log(\frac{f(x_0) - f^*}{\epsilon}) \quad (7.30)$$

$$T_\epsilon^{imp} = \frac{\sum_{i=1}^{d} \beta_i}{\lambda} log(\frac{f(x_0) - f^*}{\epsilon}) = \frac{d}{\lambda} log(\frac{f(x_0) - f^*}{\epsilon}) \quad (7.31)$$

Notice T_ϵ^{unif} is d times slower than T_ϵ^{imp}.