Coordinate Descent - An introduction

Andersen Ang

Mathématique et recherche opérationnelle
UMONS, Belgium

manshun.ang@umons.ac.be Homepage: angms.science

First draft: December 28, 2017
Last update : July 25, 2019

Overview

1. Coordinate Descent
2. Variations of Coordinate Descent
   - CD with different indexing
   - BCD
   - CD with different updates
   - Inexact BCD / BSUM
3. Summary

An unconstrained optimization problem

$$\min_{\mathbf{x}} f(\mathbf{x}) = f(x_1, x_2, ..., x_n)$$

- $\mathbf{x} \in \mathbb{R}^n$ is the vector variable of f
- x_i are scalar coordinates of $\mathbf{x}$
- function $f : \mathbb{R}^n \to \mathbb{R}$ is continuous
- no further assumption on the structure of f (e.g. convex, Lipschitz, differentiability)

Coordinate Descent Algorithm

Algorithm 1 CD framework

1: OUTPUT : $\mathbf{x} \in \mathbb{R}^n$ that minimizes $f(\mathbf{x})$
2: INITIALIZATION : $\mathbf{x}^0 \in \mathbb{R}^n$
3: **for** $k = 1, 2, ...$ until convergence condition is satisfied **do**
4: **Indexing** : Pick a coordinate index i_k
5: **Updating** : Update the selected coordinate $x_{i_k}^k$ using f and the previous iterate $\mathbf{x}^k$ while holding other coordinates fix :

$$x_j^k = \begin{cases} \text{Update}(f, \mathbf{x}^{k-1}) & \text{if } j = i_k \\ x_j^{k-1} & \text{if } j \neq i_k \end{cases}$$

6: **end for**

The Update$(f, \mathbf{x}^{k-1})$ itself is to solve a optimization sub-problem. So CD itself is not really an algorithm but an conceptual algorithmic *framework*.

The Update$(f, \mathbf{x}^{k-1})$

It is to renew the selected component x_{i_k} by minimizing the objective function f with respect to x_{i_k} while holding all other coordinates fix

$$x_{i_k}^k = \arg\min_{x_{i_k}} f(x_1^{k-1}, x_2^{k-1}, \ldots, x_{i_k-1}^{k-1}, x_{i_k}, x_{i_k+1}^{k-1}, ..., x_n^{k-1})$$

Notational rearrange,

$$x_{i_k}^k = \arg\min_{x_{i_k}} f(x_{i_k} \;;\; \underbrace{x_1^{k-1}, x_2^{k-1}, \ldots, x_{i_k-1}^{k-1}, x_{i_k+1}^{k-1}, ..., x_n^{k-1}}_{\mathbf{x}_{\neq i_k}^{k-1}})$$

Short hand notation

$$x_{i_k}^k = \arg\min_{x_{i_k}} f(x_{i_k}; \mathbf{x}_{\neq i_k}^{k-1})$$

A short summary

Algorithm 2 CD framework for solving $\min_{\mathbf{x}} f(\mathbf{x})$

1: INITIALIZATION : $\mathbf{x}^0 \in \mathbb{R}^n$
2: **for** $k = 1, 2, \ldots$ until convergence condition is satisfied **do**
3: **Indexing** : Pick the coordinate index i_k
4: **Updating** : $x_{i_k} = \text{Update}(f, \mathbf{x}^{k-1}) = \arg\min_{x_{i_k}} f(x_{i_k}; \mathbf{x}^{k-1}_{\neq i_k})$
5: **end for**

CD is an conceptual framework on design of algorithm to solve optimization problem. Thus variations can be introduced on

- Indexing : the way to select i_k
- Updating : the way to formulate $\text{Update}(f, \mathbf{x}^{k-1})$

The next pages will be on the variations of CD.

Variations of CD : cyclic indexing

Algorithm 3 CD framework for solving $\min_{\mathbf{x}} f(\mathbf{x})$

1: INITIALIZATION : $\mathbf{x}^0 \in \mathbb{R}^n$
2: **for** $k = 1, 2, ...$ until convergence condition is satisfied **do**
3: Cyclic indexing : $i_{k+1} = (k \mod n) + 1$
4: **Updating** : $x_{i_k} = \text{Update}(f, \mathbf{x}^{k-1}) = \arg\min_{x_{i_k}} f(x_{i_k}; \mathbf{x}^{k-1}_{\neq i_k})$
5: **end for**

- simple indexing scheme
- what it does : select index in cyclic manner

$$i_k = \underbrace{1, 2, 3, \dots, n}_{\text{one cycle}}, \underbrace{1, 2, 3, \dots, n}_{\text{one cycle}}, \dots$$

- In general : as long as it cycle through all indices then ok. Can be irregular as :

$$i_k = \underbrace{7, 2, 3, \dots, 6, 8, \dots, n, 1}_{\text{one cycle}}, \underbrace{7, 2, 3, \dots, 6, 8, \dots, n, 1}_{\text{one cycle}}, \dots$$

Variations of CD : random indexing

Algorithm 4 CD framework for solving $\min_{\mathbf{x}} f(\mathbf{x})$

1: INITIALIZATION : $\mathbf{x}^0 \in \mathbb{R}^n$
2: **for** $k = 1, 2, \ldots$ until convergence condition is satisfied **do**
3: Random indexing : pick i_k according to $\mathbb{P}(i_k = j) = p_j$, where $\{p_i\}_{i=1}^n$ are some assigned probability
4: **Updating** : $x_{i_k} = \text{Update}(f, \mathbf{x}^{k-1}) = \arg\min_{x_{i_k}} f(x_{i_k}; \mathbf{x}^{k-1}_{\neq i_k})$
5: **end for**

Ways to assign probability :

- Uniform : index is chosen with equal chance $p_j = \dfrac{1}{n}$
- Importance : important coordinate has a higher chance being selected. "Importance" can be defined in various ways. For example, p_j can be defined as the portion of the coordinate-wise Lipschitz constant among all coordinate $p_j = \dfrac{L_j}{\sum_j L_j}$ (in fact, this indexing achieve a faster convergence than using uniform sampling if any L_i differes)

Algorithm 5 CD framework for solving $\min_{\mathbf{x}} f(\mathbf{x})$

1: INITIALIZATION : $\mathbf{x}^0 \in \mathbb{R}^n$
2: **for** $k = 1, 2, \ldots$ until convergence condition is satisfied **do**
3: Pick i_k greedily
4: **Updating** : $x_{i_k} = \text{Update}(f, \mathbf{x}^{k-1}) = \arg\min_{x_{i_k}} f(x_{i_k}; \mathbf{x}^{k-1}_{\neq i_k})$
5: **end for**

Greedy can be defined by maximum improvement

$$i_k = \arg\min_j f(x_j, \mathbf{x}^{k-1}_{\neq j})$$

or, if the function is differentiable(non-differentiable), Greedy can be defined as picking the index with largest gradient(sub-gradient)

$$i_k = \arg\min_j \|\nabla_j f(\mathbf{x}^{k-1})\|$$

or gradient normalized by Lipschitz constant

$$i_k = \arg\min_j \frac{\|\nabla_j f(\mathbf{x}^{k-1})\|}{\sqrt{L_j}}$$

If variable $\mathbf{x}$ is decomposed into s blocks that each block $\mathbf{x}_i$ is a collections of coordinate, then CD becomes BCD, which share the same algorithmic structure as CD :

Algorithm 6 CD framework for solving $\min_{\mathbf{x}} f(\mathbf{x})$

1: INITIALIZATION : $\mathbf{x}^0 \in \mathbb{R}^n$
2: **for** $k = 1, 2, ...$ until convergence condition is satisfied **do**
3: **Indexing** : Pick the coordinate index $i_k \in \{1, 2, ..., s\}$
4: **Updating** : $x_{i_k} = \text{Update}(f, \mathbf{x}^{k-1}) = \arg\min_{x_{i_k}} f(x_{i_k}; \mathbf{x}^{k-1}_{\neq i_k})$
5: **end for**

Difference between BCD and CD :

- CD on *coordinate* (scalar component of $\mathbf{x}$)
- BCD on *block of coordinate* (vector component of $\mathbf{x}$)
- n coordinates in $\mathbf{x}$ for CD, s blocks for BCD

CD can be seen as a special case of BCD.

Variations of CD : coordinate minimization update

Now consider the variation on updating. The coordinate minimization update is :

Algorithm 7 CD framework for solving $\min_{\mathbf{x}} f(\mathbf{x})$

1: INITIALIZATION : $\mathbf{x}^0 \in \mathbb{R}^n$
2: **for** $k = 1, 2, ...$ until convergence condition is satisfied **do**
3: **Indexing** : Pick the coordinate index $i_k \in \{1, 2, ..., s\}$
4: **Updating** : $\mathbf{x}_{i_k} = \arg\min_{\mathbf{x}_{i_k}} f(\mathbf{x}_{i_k}; \mathbf{x}^{k-1}_{\neq i_k})$
5: **end for**

Line 4 itself is also an optimization problem : if (computable) close form solution exists for line 4, done.
Otherwise numerical optimization method such as gradient descent can be applied on line 4 :

$$\mathbf{x}_{i_k} \leftarrow \mathbf{x}_{i_k} - t_{i_k} \nabla_{i_k} f(\mathbf{x}^{k-1}),$$

t_{i_k} step size, ∇_{i_k} partial gradient. Such CD framework is called *coordinate gradient descent* which requires $f(\mathbf{x}_{i_k}; \mathbf{x}^{k-1}_{\neq i_k})$ to be differentiable. If f is not differentiable, we get *coordinate sub-gradient algorithm* by replacing ∇f with sub-gradient.

Variations of CD : proximal point update

Adding a quadratic term on the sub-problem of coordinate minimization update, we get *coordinate proximal point algorithm* :

Algorithm 8 CD framework for solving $\min_{\mathbf{x}} f(\mathbf{x})$

1: INITIALIZATION : $\mathbf{x}^0 \in \mathbb{R}^n$
2: **for** $k = 1, 2, ...$ until convergence condition is satisfied **do**
3: **Indexing** : Pick the coordinate index $i_k \in \{1, 2, ..., s\}$
4: **Updating** : $\mathbf{x}_{i_k} = \arg\min_{\mathbf{x}_{i_k}} f(\mathbf{x}_{i_k}; \mathbf{x}_{\neq i_k}^{k-1}) + \frac{1}{2\alpha_{i_k}^{k-1}} \|\mathbf{x}_{i_k} - \mathbf{x}_{i_k}^{-}\|_2^2$
5: **end for**

- $\mathbf{x}_{i_k}^{-}$ the previous iterate of $\mathbf{x}_{i_k}$
- $\|\mathbf{x}_{i_k} - \mathbf{x}_{i_k}^{-}\|_2^2$ the proximal term
- α_{i_k} a positive constant (proximal point parameter)

The addition of the quadratic proximal term "gives" certain advantage for solving the sub-problem.
e.g. if f not differentiable/smooth, the addition of the proximal term (with a suitable α_{i_k}) makes it differentiable/smoother

Variations of CD : proximal gradient update

For structured f as $f(\mathbf{x}) = \underbrace{g(\mathbf{x})}_{\text{differentiable}} + \underbrace{h(\mathbf{x})}_{\text{non-differentiable}}$, proximal gradient update can be used :

Algorithm 9 CD framework for solving $\min_{\mathbf{x}} f(\mathbf{x})$

1: INITIALIZATION : $\mathbf{x}^0 \in \mathbb{R}^n$
2: **for** $k = 1, 2, ...$ until convergence condition is satisfied **do**
3: **Indexing** : Pick the coordinate index $i_k \in \{1, 2, ..., s\}$
4: **Updating** :

$$\mathbf{x}_{i_k} = \arg\min_{\mathbf{x}_{i_k}} g(\mathbf{x}^{k-1}) + \left\langle \nabla_{i_k} g(\mathbf{x}_{i_k}; \mathbf{x}^{k-1}_{\neq i_k}), \mathbf{x}_{i_k} - \mathbf{x}^{-}_{i_k} \right\rangle + \frac{1}{2\alpha^{k-1}_{i_k}} \|\mathbf{x}_{i_k} - \mathbf{x}^{-}_{i_k}\|^2_2 + h(\mathbf{x}_{i_k})$$

5: **end for**

What it does : minimizes the local quadratic model of $g(\mathbf{x})$ plus the non-differentiable term $h(\mathbf{x})$.

Variations of CD : other updates

Algorithm 10 CD framework for solving $\min_{\mathbf{x}} f(\mathbf{x})$

1: INITIALIZATION : $\mathbf{x}^0 \in \mathbb{R}^n$
2: **for** $k = 1, 2, \dots$ until convergence condition is satisfied **do**
3: **Indexing** : Pick the coordinate index $i_k \in \{1, 2, \dots, s\}$
4: **Updating** : $\mathbf{x}_{i_k} =$ Update$(f, \mathbf{x}^{k-1})$
5: **end for**

As CD an algoritmic framework, other updates can be applied (depends on the structure of f). For examples

- second order methods (if Hessian of f is "computable")
- dual method (if dual problem is easier to solve)
- primal-dual methods (if dual problem is easier to solve)
- ADMM
- etc.

In these cases we get *coordinate Newton*, *coordinate Dual ascent*, Coordinate primal-dual algorithm, *coordinate ADMM*.

Variations of CD : inexact BCD / BSUM

Algorithm 11 CD framework for solving $\min_{\mathbf{x}} f(\mathbf{x})$

1: INITIALIZATION : $\mathbf{x}^0 \in \mathbb{R}^n$
2: **for** $k = 1, 2, ...$ until convergence condition is satisfied **do**
3: **Indexing** : Pick the coordinate index $i_k \in \{1, 2, ..., s\}$
4: **Updating** : $\mathbf{x}_{i_k} = \text{Update}(f, \mathbf{x}^{k-1})$
5: **end for**

Suppose the sup-problem in line 4 cannot be solved easily : coordinate minimization, coordinate gradient descent, proximal point update and proximal gradient update mentioned before are not easily applicable to the problem f.

As CD is an algorithmic framework, one can consider incorporating the idea of *Majorization Minimization* here and get *inexact BCD/BSUM*.

BSUM (Block Successive Upper Bound Minimization)

Idea : as $\mathbf{x}_{i_k} = \text{Update}(f, \mathbf{x}^{k-1})$ is not "friendly", so instead of working on f, we construct a surrogate/majorizer/upper bound of f, denoted as u.

We work on the minimization of u, then use u to update f.

If f is non-convex but u is convex, we can say such approach is a *convex relaxation* : as the original non-convex f is now relaxed to a convex u.

Price to pay by relaxation : *relaxation gap* $u - f$. Normally after each time u is minimized, the upper bound u is updated to reduce the gap.

Such "relax-update-modify" approach is carried out successively, thus the framework is called SUM : Successive Upper Bound Minimization.

As we are not working on the orignal function f but an upper bound, thus this framework is also called *inexact BCD*.

BSUM (Block Successive Upper Bound Minimization)

The optimization problem :

$$\min_{\mathbf{x}} f(\mathbf{x})$$

with f not so "user-friendly".

Algorithm 12 Inexact BCD / BSUM

1: INITIALIZATION : $\mathbf{x} \in \mathbb{R}^n$
2: **for** $k = 1, 2, ...$ until convergence condition is satisfied **do**
3: **Indexing** : Pick the coordinate index $i_k \in \{1, 2, ..., s\}$
4: **Relax** : Construct an upper bound u
5: **Updating** : $\mathbf{x}_{i_k} =$ Update$(u, \mathbf{x}^{k-1})$
6: **Modify** : Modify the upper bound $u(x)$ to reduce the relaxation gap
7: **end for**

There are some requirements on the construction of the upper bound u to ensure convergence, which are out of the scope here.

Last page - summary

Introduced :

- Coordinate Descent in the most fundamental form
- Block Coordinate Descent
- Some variations on BCD such as indexing and updating
- Inexact BCD/BSUM

Not discussed :

- The convergence of BCD with various indexing and updating
- How to select which indexing and updating scheme to use
- Acceleration of CD
- Application of BCD and inexact BCD

End of document