Optimization for Machine Learning

Lecture 12: Coordinate Descent, BCD, Altmin

6.881: MIT

Suvrit Sra
Massachusetts Institute of Technology

01 Apr, 2021

Coordinate descent

$$\boxed{\text{So far: } \min f(x) = \sum_{i=1}^{n} f_i(x)}$$

Coordinate descent

So far: $\min f(x) = \sum_{i=1}^{n} f_i(x)$

Since $x \in \mathbb{R}^d$, now consider

$$\min f(x) = f(x_1, x_2, \ldots, x_d)$$

Previously, we went through $f_1, \ldots, f_n$

What if we now go through $x_1, \ldots, x_d$ one by one?

Coordinate descent

So far: $\min f(x) = \sum_{i=1}^{n} f_i(x)$

Since $x \in \mathbb{R}^d$, now consider

$$\min f(x) = f(x_1, x_2, \ldots, x_d)$$

Previously, we went through $f_1, \ldots, f_n$

What if we now go through $x_1, \ldots, x_d$ one by one?

Explore: Going through both $[n]$ and $[d]$?

Coordinate descent

Coordinate descent

- For $k = 0, 1, \ldots$

Coordinate descent

- For $k = 0, 1, \ldots$
  - Pick an index i from $\{1, \ldots, d\}$

Coordinate descent

- For $k = 0, 1, \ldots$
  - Pick an index i from $\{1, \ldots, d\}$
  - Optimize the ith coordinate

$$x_i^{k+1} \leftarrow \operatorname*{argmin}_{\xi \in \mathbb{R}} f(\underbrace{x_1^{k+1}, \ldots, x_{i-1}^{k+1}}_{\text{done}}, \underbrace{\xi}_{\text{current}}, \underbrace{x_{i+1}^k, \ldots, x_d^k}_{\text{todo}})$$

Coordinate descent

- For $k = 0, 1, \ldots$
  - Pick an index i from $\{1, \ldots, d\}$
  - Optimize the ith coordinate

$$x_i^{k+1} \leftarrow \operatorname*{argmin}_{\xi \in \mathbb{R}} f(\underbrace{x_1^{k+1}, \ldots, x_{i-1}^{k+1}}_{\text{done}}, \underbrace{\xi}_{\text{current}}, \underbrace{x_{i+1}^{k}, \ldots, x_d^{k}}_{\text{todo}})$$

- Decide when/how to stop; *return* x^k

Coordinate descent - context

♣ One of the simplest optimization methods

Coordinate descent - context

- One of the simplest optimization methods
- Old idea: Gauss-Seidel, Jacobi methods for linear systems!

Coordinate descent - context

- ♣ One of the simplest optimization methods
- ♣ Old idea: Gauss-Seidel, Jacobi methods for linear systems!
- ♣ Can be "slow", but sometimes very competitive

Coordinate descent - context

- One of the simplest optimization methods
- Old idea: Gauss-Seidel, Jacobi methods for linear systems!
- Can be “slow”, but sometimes very competitive
- Gradient, subgradient, incremental methods also “slow”

Coordinate descent - context

- One of the simplest optimization methods
- Old idea: Gauss-Seidel, Jacobi methods for linear systems!
- Can be "slow", but sometimes very competitive
- Gradient, subgradient, incremental methods also "slow"
- But incremental, stochastic gradient methods are scalable

Coordinate descent - context

- ♣ One of the simplest optimization methods
- ♣ Old idea: Gauss-Seidel, Jacobi methods for linear systems!
- ♣ Can be "slow", but sometimes very competitive
- ♣ Gradient, subgradient, incremental methods also "slow"
- ♣ But incremental, stochastic gradient methods are scalable
- ♣ Renewed interest in CD was driven by ML

Coordinate descent - context

- ♣ One of the simplest optimization methods
- ♣ Old idea: Gauss-Seidel, Jacobi methods for linear systems!
- ♣ Can be “slow”, but sometimes very competitive
- ♣ Gradient, subgradient, incremental methods also “slow”
- ♣ But incremental, stochastic gradient methods are scalable
- ♣ Renewed interest in CD was driven by ML
- ♣ Notice: in general CD is “derivative free”

CD – which coordinate?

CD – which coordinate?

Gauss-Southwell: If f is differentiable, at iteration k, pick the index that minimizes $[\nabla f(x_k)]_i$

CD – which coordinate?

Gauss-Southwell: If f is differentiable, at iteration k, pick the index that minimizes $[\nabla f(x_k)]_i$

Derivative free rules:

CD – which coordinate?

Gauss-Southwell: If f is differentiable, at iteration k, pick the index that minimizes $[\nabla f(x_k)]_i$

Derivative free rules:

♣ **Cyclic** order $1, 2, \ldots, d, 1, \ldots$

CD – which coordinate?

Gauss-Southwell: If f is differentiable, at iteration k, pick the index that minimizes $[\nabla f(x_k)]_i$

Derivative free rules:

- ♣ **Cyclic** order $1, 2, \ldots, d, 1, \ldots$
- ♣ **Almost cyclic:** Each coordinate $1 \leq i \leq d$ picked at least once every B successive iterations ($B \geq d$)

CD – which coordinate?

Gauss-Southwell: If f is differentiable, at iteration k, pick the index that minimizes $[\nabla f(x_k)]_i$

Derivative free rules:

- ♣ **Cyclic** order $1, 2, \ldots, d, 1, \ldots$
- ♣ **Almost cyclic:** Each coordinate $1 \le i \le d$ picked at least once every B successive iterations ($B \ge d$)
- ♣ **Double sweep**, $1, \ldots, d$ then $d-1, \ldots, 1$, repeat

CD – which coordinate?

Gauss-Southwell: If f is differentiable, at iteration k, pick the index that minimizes $[\nabla f(x_k)]_i$

Derivative free rules:

- **Cyclic** order $1, 2, \ldots, d, 1, \ldots$
- **Almost cyclic:** Each coordinate $1 \le i \le d$ picked at least once every B successive iterations ($B \ge d$)
- **Double sweep**, $1, \ldots, d$ then $d-1, \ldots, 1$, repeat
- **Cylic with permutation**: random order each cycle

CD – which coordinate?

Gauss-Southwell: If f is differentiable, at iteration k, pick the index that minimizes $[\nabla f(x_k)]_i$

Derivative free rules:

- ♣ **Cyclic** order $1, 2, \ldots, d, 1, \ldots$
- ♣ **Almost cyclic:** Each coordinate $1 \leq i \leq d$ picked at least once every B successive iterations ($B \geq d$)
- ♣ **Double sweep**, $1, \ldots, d$ then $d-1, \ldots, 1$, repeat
- ♣ **Cylic with permutation**: random order each cycle
- ♣ **Random sampling**: pick random index at each iteration

Which ones would you prefer? Why?

Exercise: CD for least squares

$$\min_x \; \|Ax - b\|_2^2$$

Exercise: Obtain an update for j-th coordinate

Coordinate descent update

$$x_j \leftarrow \frac{\sum_{i=1}^m a_{ij}\left(b_i - \sum_{l \neq j} a_{il} x_l\right)}{\sum_{i=1}^m a_{ij}^2}$$

(dropped superscripts, since we overwrite)

Coordinate descent – some remarks

Advantages

- ♢ Each iteration usually cheap (single variable optimization)

Coordinate descent – some remarks

Advantages

- ♢ Each iteration usually cheap (single variable optimization)
- ♢ No extra storage vectors needed

Coordinate descent – some remarks

Advantages

- ♢ Each iteration usually cheap (single variable optimization)
- ♢ No extra storage vectors needed
- ♢ **No stepsize tuning**

Coordinate descent – some remarks

Advantages

- ♢ Each iteration usually cheap (single variable optimization)
- ♢ No extra storage vectors needed
- ♢ **No stepsize tuning** 🙂
- ♢ No other pesky parameters (usually) that must be tuned

Coordinate descent – some remarks

Advantages

- ♢ Each iteration usually cheap (single variable optimization)
- ♢ No extra storage vectors needed
- ♢ **No stepsize tuning** 🙂
- ♢ No other pesky parameters (usually) that must be tuned
- ♢ Simple to implement

Coordinate descent – some remarks

Advantages

- ♢ Each iteration usually cheap (single variable optimization)
- ♢ No extra storage vectors needed
- ♢ **No stepsize tuning** 🙂
- ♢ No other pesky parameters (usually) that must be tuned
- ♢ Simple to implement
- ♢ Can work well for large-scale problems

Coordinate descent – some remarks

Advantages

- ♢ Each iteration usually cheap (single variable optimization)
- ♢ No extra storage vectors needed
- ♢ **No stepsize tuning** ☺
- ♢ No other pesky parameters (usually) that must be tuned
- ♢ Simple to implement
- ♢ Can work well for large-scale problems

Disadvantages

- ♠ Tricky if single variable optimization is hard

Coordinate descent – some remarks

Advantages

- ♢ Each iteration usually cheap (single variable optimization)
- ♢ No extra storage vectors needed
- ♢ **No stepsize tuning** ☺
- ♢ No other pesky parameters (usually) that must be tuned
- ♢ Simple to implement
- ♢ Can work well for large-scale problems

Disadvantages

- ♠ Tricky if single variable optimization is hard
- ♠ Convergence theory can be complicated

Coordinate descent – some remarks

Advantages

- ♢ Each iteration usually cheap (single variable optimization)
- ♢ No extra storage vectors needed
- ♢ **No stepsize tuning** ☺
- ♢ No other pesky parameters (usually) that must be tuned
- ♢ Simple to implement
- ♢ Can work well for large-scale problems

Disadvantages

- ♠ Tricky if single variable optimization is hard
- ♠ Convergence theory can be complicated
- ♠ Can slow down near optimum

Coordinate descent – some remarks

Advantages

- ♢ Each iteration usually cheap (single variable optimization)
- ♢ No extra storage vectors needed
- ♢ **No stepsize tuning** 🙂
- ♢ No other pesky parameters (usually) that must be tuned
- ♢ Simple to implement
- ♢ Can work well for large-scale problems

Disadvantages

- ♠ Tricky if single variable optimization is hard
- ♠ Convergence theory can be complicated
- ♠ Can slow down near optimum
- ♠ Nonsmooth case more tricky

Coordinate descent – some remarks

Advantages

- ◇ Each iteration usually cheap (single variable optimization)
- ◇ No extra storage vectors needed
- ◇ **No stepsize tuning** 🙂
- ◇ No other pesky parameters (usually) that must be tuned
- ◇ Simple to implement
- ◇ Can work well for large-scale problems

Disadvantages

- ♠ Tricky if single variable optimization is hard
- ♠ Convergence theory can be complicated
- ♠ Can slow down near optimum
- ♠ Nonsmooth case more tricky
- ♠ **Explore:** not easy to use for deep learning...

BCD

(Basics, Convergence)

Block coordinate descent (BCD)

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) := f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) \\ & \boldsymbol{x} \in \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_m. \end{aligned}$$

Block coordinate descent (BCD)

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) := f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) \\ & \boldsymbol{x} \in \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_m. \end{aligned}$$

Gauss-Seidel update

$$\boldsymbol{x}_i^{k+1} \leftarrow \operatorname*{argmin}_{\boldsymbol{\xi} \in \mathcal{X}_i} f(\underbrace{\boldsymbol{x}_1^{k+1}, \ldots, \boldsymbol{x}_{i-1}^{k+1}}_{\textbf{done}}, \underbrace{\boldsymbol{\xi}}_{\textbf{current}}, \underbrace{\boldsymbol{x}_{i+1}^{k}, \ldots, \boldsymbol{x}_m^{k}}_{\textbf{todo}})$$

Block coordinate descent (BCD)

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) := f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) \\ & \boldsymbol{x} \in \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_m. \end{aligned}$$

Gauss-Seidel update

$$\boldsymbol{x}_i^{k+1} \leftarrow \operatorname*{argmin}_{\boldsymbol{\xi} \in \mathcal{X}_i} f(\underbrace{\boldsymbol{x}_1^{k+1}, \ldots, \boldsymbol{x}_{i-1}^{k+1}}_{\text{done}}, \underbrace{\boldsymbol{\xi}}_{\text{current}}, \underbrace{\boldsymbol{x}_{i+1}^{k}, \ldots, \boldsymbol{x}_m^{k}}_{\text{todo}})$$

Jacobi update (easy to parallelize)

$$\boldsymbol{x}_i^{k+1} \leftarrow \operatorname*{argmin}_{\boldsymbol{\xi} \in \mathcal{X}_i} f(\underbrace{\boldsymbol{x}_1^{k}, \ldots, \boldsymbol{x}_{i-1}^{k}}_{\text{don't clobber}}, \underbrace{\boldsymbol{\xi}}_{\text{current}}, \underbrace{\boldsymbol{x}_{i+1}^{k}, \ldots, \boldsymbol{x}_m^{k}}_{\text{todo}})$$

BCD – convergence

Theorem. Let f be C^1 over $\mathcal{X} := \prod_{i=1}^m \mathcal{X}_i$. Assume for each block i and $\boldsymbol{x} \in \mathcal{X}$, the minimum

$$\min_{\boldsymbol{\xi} \in \mathcal{X}_i} f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i+1}, \boldsymbol{\xi}, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_m)$$

is **uniquely attained**. Then, every limit point of the sequence $\{x^k\}$ generated by BCD, is a stationary point of f.

BCD – convergence

Theorem. Let f be C^1 over $\mathcal{X} := \prod_{i=1}^m \mathcal{X}_i$. Assume for each block i and $\boldsymbol{x} \in \mathcal{X}$, the minimum

$$\min_{\boldsymbol{\xi} \in \mathcal{X}_i} f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i+1}, \boldsymbol{\xi}, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_m)$$

is **uniquely attained**. Then, every limit point of the sequence $\{\boldsymbol{x}^k\}$ generated by BCD, is a stationary point of f.

Corollary. If f is in addition convex, then every limit point of the BCD sequence $\{\boldsymbol{x}^k\}$ is a global minimum.

BCD – convergence

Theorem. Let f be C^1 over $\mathcal{X} := \prod_{i=1}^m \mathcal{X}_i$. Assume for each block i and $\boldsymbol{x} \in \mathcal{X}$, the minimum

$$\min_{\boldsymbol{\xi} \in \mathcal{X}_i} f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i+1}, \boldsymbol{\xi}, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_m)$$

is **uniquely attained**. Then, every limit point of the sequence $\{\boldsymbol{x}^k\}$ generated by BCD, is a stationary point of f.

Corollary. If f is in addition convex, then every limit point of the BCD sequence $\{\boldsymbol{x}^k\}$ is a global minimum.

- **Unique solutions** of subproblems not always possible
- Above result is only **asymptotic** (holds in the limit)
- **Warning!** BCD may cycle indefinitely without converging, if number blocks > 2 and objective nonconvex.

BCD – Two blocks

Two block BCD

$$\text{minimize} f(\boldsymbol{x}) = f(\boldsymbol{x}_1, \boldsymbol{x}_2) \quad \boldsymbol{x} \in \mathcal{X}_1 \times \mathcal{X}_2.$$

BCD – Two blocks

Two block BCD

$$\text{minimize} f(\boldsymbol{x}) = f(\boldsymbol{x}_1, \boldsymbol{x}_2) \quad \boldsymbol{x} \in \mathcal{X}_1 \times \mathcal{X}_2.$$

Theorem. (Grippo & Sciandrone (2000)). Let f be continuously differentiable. Let $\mathcal{X}_1, \mathcal{X}_2$ be closed and convex. Assume both BCD subproblems have solutions and the sequence $\{\boldsymbol{x}^k\}$ has limit points. Then, every limit point of $\{\boldsymbol{x}^k\}$ is stationary.

BCD – Two blocks

Two block BCD

$$\text{minimize} f(\boldsymbol{x}) = f(\boldsymbol{x}_1, \boldsymbol{x}_2) \quad \boldsymbol{x} \in \mathcal{X}_1 \times \mathcal{X}_2.$$

Theorem. (Grippo & Sciandrone (2000)). Let f be continuously differentiable. Let $\mathcal{X}_1, \mathcal{X}_2$ be closed and convex. Assume both BCD subproblems have solutions and the sequence $\{\boldsymbol{x}^k\}$ has limit points. Then, every limit point of $\{\boldsymbol{x}^k\}$ is stationary.

- No need of **unique solutions** to subproblems
- BCD for 2 blocks is also called: **Alternating Minimization**

CD – projection onto convex sets

$$\begin{aligned} \min \quad & \tfrac{1}{2}\|x-y\|_2^2 \\ \text{s.t.} \quad & x \in C_1 \cap C_2 \cap \cdots \cap C_m. \end{aligned}$$

CD – projection onto convex sets

$$\begin{aligned} \min \quad & \tfrac{1}{2}\|x - y\|_2^2 \\ \text{s.t.} \quad & x \in C_1 \cap C_2 \cap \cdots \cap C_m. \end{aligned}$$

Solution 1: Rewrite using indicator functions

$$\min \quad \tfrac{1}{2}\|x - y\|_2^2 + \sum\nolimits_{i=1}^{m} \delta_{C_i}(x).$$

- ▶ Now invoke Douglas-Rachford using the product-space trick

CD – projection onto convex sets

$$\begin{aligned} \min \quad & \tfrac{1}{2}\|x-y\|_2^2 \\ \text{s.t.} \quad & x \in C_1 \cap C_2 \cap \cdots \cap C_m. \end{aligned}$$

Solution 1: Rewrite using indicator functions

$$\min \quad \tfrac{1}{2}\|x-y\|_2^2 + \sum\nolimits_{i=1}^{m} \delta_{C_i}(x).$$

▶ Now invoke Douglas-Rachford using the product-space trick

Solution 2: Take dual of the above formulation

Solution 1: Product space technique

- Original problem over $\mathcal{H} = \mathbb{R}^n$

Solution 1: Product space technique

- Original problem over $\mathcal{H} = \mathbb{R}^n$
- Suppose we have $\sum_{i=1}^{n} f_i(x)$

Solution 1: Product space technique

- Original problem over $\mathcal{H} = \mathbb{R}^n$
- Suppose we have $\sum_{i=1}^{n} f_i(x)$
- Introduce n new variables $(x_1, \ldots, x_n)$

Solution 1: Product space technique

- Original problem over $\mathcal{H} = \mathbb{R}^n$
- Suppose we have $\sum_{i=1}^n f_i(x)$
- Introduce n new variables $(x_1, \ldots, x_n)$
- Now problem is over domain $\mathcal{H}^n := \times_{i=1}^n \mathcal{H}$

Solution 1: Product space technique

- Original problem over $\mathcal{H} = \mathbb{R}^n$
- Suppose we have $\sum_{i=1}^{n} f_i(x)$
- Introduce n new variables $(x_1, \ldots, x_n)$
- Now problem is over domain $\mathcal{H}^n := \bigtimes_{i=1}^{n} \mathcal{H}$
- New constraint: $x_1 = x_2 = \ldots = x_n$

$$\begin{aligned} \min_{(x_1,\ldots,x_n)} \quad & \sum\nolimits_i f_i(x_i) \\ \text{s.t.} \quad & x_1 = x_2 = \cdots = x_n. \end{aligned}$$

Technique due to: G. Pierra (1976)

Solution 1: Product space technique

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) + \mathbb{1}_{\mathcal{B}}(\boldsymbol{x})$$

where $\boldsymbol{x} \in \mathcal{H}^n$ and $\mathcal{B} = \{\boldsymbol{z} \in \mathcal{H}^n \mid \boldsymbol{z} = (x, x, \ldots, x)\}$

Solution 1: Product space technique

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) + \mathbb{1}_{\mathcal{B}}(\boldsymbol{x})$$

where $\boldsymbol{x} \in \mathcal{H}^n$ and $\mathcal{B} = \{\boldsymbol{z} \in \mathcal{H}^n \mid \boldsymbol{z} = (x, x, \ldots, x)\}$

- Let $\boldsymbol{y} = (y_1, \ldots, y_n)$

Solution 1: Product space technique

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) + \mathbb{1}_{\mathcal{B}}(\boldsymbol{x})$$

where $\boldsymbol{x} \in \mathcal{H}^n$ and $\mathcal{B} = \{\boldsymbol{z} \in \mathcal{H}^n \mid \boldsymbol{z} = (x, x, \ldots, x)\}$

- Let $\boldsymbol{y} = (y_1, \ldots, y_n)$
- $\operatorname{prox}_f(\boldsymbol{y}) = (\operatorname{prox}_{f_1}(y_1), \ldots, \operatorname{prox}_{f_n}(y_n))$

Solution 1: Product space technique

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) + \mathbb{1}_{\mathcal{B}}(\boldsymbol{x})$$

where $\boldsymbol{x} \in \mathcal{H}^n$ and $\mathcal{B} = \{\boldsymbol{z} \in \mathcal{H}^n \mid \boldsymbol{z} = (x, x, \ldots, x)\}$

- Let $\boldsymbol{y} = (y_1, \ldots, y_n)$
- $\text{prox}_f(\boldsymbol{y}) = (\text{prox}_{f_1}(y_1), \ldots, \text{prox}_{f_n}(y_n))$
- $\text{prox}_{\mathcal{B}} \equiv \Pi_{\mathcal{B}}(\boldsymbol{y})$ can be solved as follows:

Solution 1: Product space technique

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) + \mathbb{1}_{\mathcal{B}}(\boldsymbol{x})$$

where $\boldsymbol{x} \in \mathcal{H}^n$ and $\mathcal{B} = \{\boldsymbol{z} \in \mathcal{H}^n \mid \boldsymbol{z} = (x, x, \ldots, x)\}$

- Let $\boldsymbol{y} = (y_1, \ldots, y_n)$
- $\operatorname{prox}_f(\boldsymbol{y}) = (\operatorname{prox}_{f_1}(y_1), \ldots, \operatorname{prox}_{f_n}(y_n))$
- $\operatorname{prox}_{\mathcal{B}} \equiv \Pi_{\mathcal{B}}(\boldsymbol{y})$ can be solved as follows:

$$\begin{aligned} \min\nolimits_{z \in \mathcal{B}} &\quad \tfrac{1}{2}\|\boldsymbol{z} - \boldsymbol{y}\|_2^2 \\ \min\nolimits_{x \in \mathcal{H}} &\quad \textstyle\sum_i \tfrac{1}{2}\|x - y_i\|_2^2 \\ \implies &\quad x = \tfrac{1}{n}\textstyle\sum_i y_i \end{aligned}$$

Exercise: Work out the details of the Douglas-Rachford algorithm using the above product space trick.

Remark: This technique commonly exploited in ADMM too

Solution 2: proximal Dykstra

$$\boxed{\min \quad \frac{1}{2}\|x - y\|_2^2 + f(x) + h(x)}$$

Solution 2: proximal Dykstra

$$\boxed{\min \quad \tfrac{1}{2}\|x-y\|_2^2 + f(x) + h(x)}$$

$$L(x,z,w,\nu,\mu) := \tfrac{1}{2}\|x-y\|_2^2 + f(z) + h(w) + \nu^T(x-z) + \mu^T(x-w)$$

$$g(\nu,\mu) \quad := \quad \inf_{x,z,w} L(x,z,\nu,\mu)$$

Solution 2: proximal Dykstra

$$\boxed{\min \quad \tfrac{1}{2}\|x-y\|_2^2 + f(x) + h(x)}$$

$$L(x,z,w,\nu,\mu) := \tfrac{1}{2}\|x-y\|_2^2 + f(z) + h(w) + \nu^T(x-z) + \mu^T(x-w)$$

$$\begin{aligned} g(\nu,\mu) \quad &:= \quad \inf_{x,z,w} L(x,z,\nu,\mu) \\ x - y + \nu + \mu = 0 \quad &\Longrightarrow \quad x = y - \nu - \mu \end{aligned}$$

Solution 2: proximal Dykstra

$$\boxed{\min \quad \tfrac{1}{2}\|x-y\|_2^2 + f(x) + h(x)}$$

$$L(x,z,w,\nu,\mu) := \tfrac{1}{2}\|x-y\|_2^2 + f(z) + h(w) + \nu^T(x-z) + \mu^T(x-w)$$

$$\begin{aligned} g(\nu,\mu) \quad &:= \quad \inf_{x,z,w} L(x,z,\nu,\mu) \\ x-y+\nu+\mu=0 \quad &\Longrightarrow \quad x = y-\nu-\mu \\ g(\nu,\mu) \quad &= \quad -\tfrac{1}{2}\|\nu+\mu\|_2^2 + (\nu+\mu)^T y - f^*(\nu) - h^*(\mu) \end{aligned}$$

Solution 2: proximal Dykstra

$$\boxed{\min \quad \tfrac{1}{2}\|x-y\|_2^2 + f(x) + h(x)}$$

$$L(x,z,w,\nu,\mu) := \tfrac{1}{2}\|x-y\|_2^2 + f(z) + h(w) + \nu^T(x-z) + \mu^T(x-w)$$

$$\begin{aligned} g(\nu,\mu) \quad &:= \quad \inf_{x,z,w} L(x,z,\nu,\mu) \\ x - y + \nu + \mu = 0 \quad &\implies \quad x = y - \nu - \mu \\ g(\nu,\mu) \quad &= \quad -\tfrac{1}{2}\|\nu+\mu\|_2^2 + (\nu+\mu)^T y - f^*(\nu) - h^*(\mu) \end{aligned}$$

Dual as minimization problem

$$\boxed{\min\ k(\nu,\mu) := \tfrac{1}{2}\|\nu+\mu-y\|_2^2 + f^*(\nu) + h^*(\mu)}$$

The Proximal-Dykstra method

Apply CD to $k(\nu, \mu) = \frac{1}{2}\|\nu + \mu - y\|_2^2 + f^*(\nu) + h^*(\mu)$

The Proximal-Dykstra method

Apply CD to $k(\nu,\mu) = \frac{1}{2}\|\nu+\mu-y\|_2^2 + f^*(\nu) + h^*(\mu)$

$$\nu_{k+1} \quad = \quad \operatorname{argmin}_{\nu}\ k(\nu,\mu_k)$$

The Proximal-Dykstra method

Apply CD to $k(\nu, \mu) = \frac{1}{2}\|\nu + \mu - y\|_2^2 + f^*(\nu) + h^*(\mu)$

$$\begin{aligned} \nu_{k+1} &= \operatorname{argmin}_\nu \; k(\nu, \mu_k) \\ \mu_{k+1} &= \operatorname{argmin}_\mu \; k(\nu_{k+1}, \mu) \end{aligned}$$

The Proximal-Dykstra method

Apply CD to $k(\nu, \mu) = \frac{1}{2}\|\nu + \mu - y\|_2^2 + f^*(\nu) + h^*(\mu)$

$$\begin{aligned} \nu_{k+1} &= \operatorname{argmin}_{\nu}\; k(\nu, \mu_k) \\ \mu_{k+1} &= \operatorname{argmin}_{\mu}\; k(\nu_{k+1}, \mu) \end{aligned}$$

- $0 \in \nu + \mu_k - y + \partial f^*(\nu)$

The Proximal-Dykstra method

Apply CD to $k(\nu, \mu) = \frac{1}{2}\|\nu + \mu - y\|_2^2 + f^*(\nu) + h^*(\mu)$

$$
\begin{aligned}
\nu_{k+1} &= \operatorname{argmin}_{\nu}\ k(\nu, \mu_k) \\
\mu_{k+1} &= \operatorname{argmin}_{\mu}\ k(\nu_{k+1}, \mu)
\end{aligned}
$$

- $0 \in \nu + \mu_k - y + \partial f^*(\nu)$
- $0 \in \nu_{k+1} + \mu - y + \partial h^*(\mu)$

The Proximal-Dykstra method

Apply CD to $k(\nu, \mu) = \frac{1}{2}\|\nu + \mu - y\|_2^2 + f^*(\nu) + h^*(\mu)$

$$\begin{aligned} \nu_{k+1} &= \operatorname{argmin}_\nu \; k(\nu, \mu_k) \\ \mu_{k+1} &= \operatorname{argmin}_\mu \; k(\nu_{k+1}, \mu) \end{aligned}$$

- $0 \in \nu + \mu_k - y + \partial f^*(\nu)$
- $0 \in \nu_{k+1} + \mu - y + \partial h^*(\mu)$
- $y - \mu_k \in \nu + \partial f^*(\nu) = (I + \partial f^*)(\nu)$
  $\implies \nu = \mathrm{prox}_{f^*}(y - \mu_k)$

The Proximal-Dykstra method

Apply CD to $k(\nu, \mu) = \frac{1}{2}\|\nu + \mu - y\|_2^2 + f^*(\nu) + h^*(\mu)$

$$\begin{aligned}\nu_{k+1} &= \operatorname{argmin}_\nu \; k(\nu, \mu_k) \\ \mu_{k+1} &= \operatorname{argmin}_\mu \; k(\nu_{k+1}, \mu)\end{aligned}$$

- $0 \in \nu + \mu_k - y + \partial f^*(\nu)$
- $0 \in \nu_{k+1} + \mu - y + \partial h^*(\mu)$
- $y - \mu_k \in \nu + \partial f^*(\nu) = (I + \partial f^*)(\nu)$
  $\implies \nu = \operatorname{prox}_{f^*}(y - \mu_k) \implies \nu = y - \mu_k - \operatorname{prox}_f(y - \mu_k)$

The Proximal-Dykstra method

Apply CD to $k(\nu, \mu) = \frac{1}{2}\|\nu + \mu - y\|_2^2 + f^*(\nu) + h^*(\mu)$

$$\begin{aligned} \nu_{k+1} &= \operatorname{argmin}_\nu\ k(\nu, \mu_k) \\ \mu_{k+1} &= \operatorname{argmin}_\mu\ k(\nu_{k+1}, \mu) \end{aligned}$$

- $0 \in \nu + \mu_k - y + \partial f^*(\nu)$
- $0 \in \nu_{k+1} + \mu - y + \partial h^*(\mu)$
- $y - \mu_k \in \nu + \partial f^*(\nu) = (I + \partial f^*)(\nu)$
  $\implies \nu = \operatorname{prox}_{f^*}(y - \mu_k) \implies \nu = y - \mu_k - \operatorname{prox}_f(y - \mu_k)$
- Similarly, we see that
  $\mu = y - \nu_{k+1} - \operatorname{prox}_h(y - \nu_{k+1})$

The Proximal-Dykstra method

Apply CD to $k(\nu,\mu) = \frac{1}{2}\|\nu+\mu-y\|_2^2 + f^*(\nu) + h^*(\mu)$

$$\begin{aligned} \nu_{k+1} &= \operatorname{argmin}_\nu\ k(\nu,\mu_k) \\ \mu_{k+1} &= \operatorname{argmin}_\mu\ k(\nu_{k+1},\mu) \end{aligned}$$

- $0 \in \nu + \mu_k - y + \partial f^*(\nu)$
- $0 \in \nu_{k+1} + \mu - y + \partial h^*(\mu)$
- $y - \mu_k \in \nu + \partial f^*(\nu) = (I + \partial f^*)(\nu)$
  $\implies \nu = \operatorname{prox}_{f^*}(y-\mu_k) \implies \nu = y - \mu_k - \operatorname{prox}_f(y-\mu_k)$
- Similarly, we see that
  $\mu = y - \nu_{k+1} - \operatorname{prox}_h(y - \nu_{k+1})$

$$\boxed{\begin{aligned} \nu_{k+1} &\leftarrow y - \mu_k - \operatorname{prox}_f(y-\mu_k) \\ \mu_{k+1} &\leftarrow y - \nu_{k+1} - \operatorname{prox}_h(y-\nu_{k+1}) \end{aligned}}$$

Proximal-Dykstra as CD

- Simplify, and use Lagrangian stationarity to obtain primal

$$x = y - \nu - \mu \implies y - \mu = x + \nu$$

Proximal-Dykstra as CD

- Simplify, and use Lagrangian stationarity to obtain primal

$$x = y - \nu - \mu \implies y - \mu = x + \nu$$

- Thus, the CD iteration may be rewritten as

$$\begin{aligned} t_k &\leftarrow \mathrm{prox}_f(x_k + \nu_k) \\ \nu_{k+1} &\leftarrow x_k + \nu_k - t_k \\ x_{k+1} &\leftarrow \mathrm{prox}_h(\mu_k + t_k) \\ \mu_{k+1} &\leftarrow \mu_k + t_k - x_{k+1} \end{aligned}$$

- We used: $\mathrm{prox}_h(y - \nu_{k+1}) = \mu_{k+1} - y - \nu_{k+1} = x_{k+1}$

Proximal-Dykstra as CD

- Simplify, and use Lagrangian stationarity to obtain primal

$$x = y - \nu - \mu \implies y - \mu = x + \nu$$

- Thus, the CD iteration may be rewritten as

$$\begin{aligned} t_k &\leftarrow \mathrm{prox}_f(x_k + \nu_k) \\ \nu_{k+1} &\leftarrow x_k + \nu_k - t_k \\ x_{k+1} &\leftarrow \mathrm{prox}_h(\mu_k + t_k) \\ \mu_{k+1} &\leftarrow \mu_k + t_k - x_{k+1} \end{aligned}$$

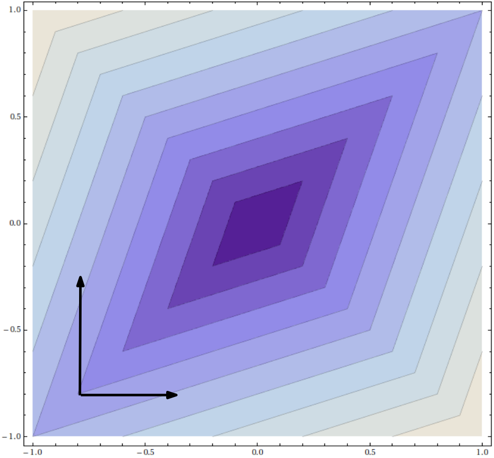
- We used: $\mathrm{prox}_h(y - \nu_{k+1}) = \mu_{k+1} - y - \nu_{k+1} = x_{k+1}$
- This is the proximal-Dykstra method!

Explore: Pros-cons of Prox-Dykstra versus product space+DR

CD – nonsmooth case

CD for nonsmooth convex problems

$$\min |x_1 - x_2| + \frac{1}{2}|x_1 + x_2|$$



CD for separable nonsmoothness

- Nonsmooth part is **separable**

$$\boxed{\min_{x\in\mathbb{R}^d} f(x) + \sum\nolimits_{i=1}^{d} r_i(x_i)}$$

CD for separable nonsmoothness

- Nonsmooth part is **separable**

$$\min_{x\in\mathbb{R}^d} f(x) + \sum\nolimits_{i=1}^{d} r_i(x_i)$$

Theorem. If f is convex, continuously differentiable, each $r_i(x)$ is closed, convex, and each coordinate admits a **unique** solution. Further, assume we go through all coordinates in an essentially cyclic way. Then, the sequence $\{x^k\}$ generated by CD is bounded, and every limit point of it is optimal.

CD for separable nonsmoothness

- Nonsmooth part is **separable**

$$\min_{x \in \mathbb{R}^d} f(x) + \sum_{i=1}^{d} r_i(x_i)$$

Theorem. If f is convex, continuously differentiable, each $r_i(x)$ is closed, convex, and each coordinate admits a **unique** solution. Further, assume we go through all coordinates in an essentially cyclic way. Then, the sequence $\{x^k\}$ generated by CD is bounded, and every limit point of it is optimal.

Remark: A related result for **nonconvex** problems with separable non-smoothness (under more assumptions), can be found in: "*Convergence of Block Coordinate Descent Method for Nondifferentiable Minimization*" by P. Tseng (2001).

CD – iteration complexity

CD non-asymptotic rate

- So far, we saw CD based on essentially cyclic rules

CD non-asymptotic rate

- So far, we saw CD based on essentially cyclic rules
- It is difficult to prove **global** convergence and almost impossible to estimate **global** rate of convergence

CD non-asymptotic rate

- So far, we saw CD based on essentially cyclic rules
- It is difficult to prove **global** convergence and almost impossible to estimate **global** rate of convergence
- Above results highlighted at best local (asymptotic) rates

CD non-asymptotic rate

- So far, we saw CD based on essentially cyclic rules
- It is difficult to prove **global** convergence and almost impossible to estimate **global** rate of convergence
- Above results highlighted at best local (asymptotic) rates
- Consider the unconstrained problem $\min f(x)$, s.t., $x \in \mathbb{R}^d$

CD non-asymptotic rate

- So far, we saw CD based on essentially cyclic rules
- It is difficult to prove **global** convergence and almost impossible to estimate **global** rate of convergence
- Above results highlighted at best local (asymptotic) rates

- Consider the unconstrained problem $\min f(x)$, s.t., $x \in \mathbb{R}^d$
- Assume f is convex, with **componentwise** Lipschitz gradients

$$|\nabla_i f(x + he_i) - \nabla_i f(x)| \leq L_i |h|, \quad x \in \mathbb{R}^d, h \in \mathbb{R}.$$

Here e_i denotes the ith canonical basis vector

CD non-asymptotic rate

- So far, we saw CD based on essentially cyclic rules
- It is difficult to prove **global** convergence and almost impossible to estimate **global** rate of convergence
- Above results highlighted at best local (asymptotic) rates

- Consider the unconstrained problem $\min f(x)$, s.t., $x \in \mathbb{R}^d$
- Assume f is convex, with **componentwise** Lipschitz gradients

$$|\nabla_i f(x + he_i) - \nabla_i f(x)| \leq L_i|h|, \quad x \in \mathbb{R}^d, h \in \mathbb{R}.$$

Here e_i denotes the ith canonical basis vector

Choose $x_0 \in \mathbb{R}^d$. Let $M = \max_i L_i$; For $k \geq 0$

$$i_k = \underset{1\leq i\leq d}{\operatorname{argmax}} |\nabla_i f(x_k)|$$

$$x_{k+1} = x_k - \tfrac{1}{M}\nabla_{i_k} f(x_k) e_{i_k}.$$

CD – non-asymptotic convergence

Theorem. Let $\{x^k\}$ be iterate sequence generated by above greedy CD method. Then,

$$f(x_k) - f^* \leq \frac{2dM\|x_0 - x^*\|_2^2}{k+4}, \quad k \geq 0.$$

CD – non-asymptotic convergence

Theorem. Let $\{x^k\}$ be iterate sequence generated by above greedy CD method. Then,

$$f(x_k) - f^* \le \frac{2dM\|x_0 - x^*\|_2^2}{k+4}, \quad k \ge 0.$$

- Looks like gradient-descent $O(1/k)$ bound for C_L^1 cvx
- Notice factor of d in the numerator!

CD – non-asymptotic convergence

Theorem. Let $\{x^k\}$ be iterate sequence generated by above greedy CD method. Then,

$$f(x_k) - f^* \le \frac{2dM\|x_0 - x^*\|_2^2}{k+4}, \quad k \ge 0.$$

- Looks like gradient-descent $O(1/k)$ bound for C_L^1 cvx
- Notice factor of d in the numerator!
- But this method is impractical

CD – non-asymptotic convergence

Theorem. Let $\{x^k\}$ be iterate sequence generated by above greedy CD method. Then,

$$f(x_k) - f^* \leq \frac{2dM\|x_0 - x^*\|_2^2}{k+4}, \quad k \geq 0.$$

- Looks like gradient-descent $O(1/k)$ bound for C_L^1 cvx
- Notice factor of d in the numerator!
- But this method is impractical
- At each step, it requires access to **full gradient**

CD – non-asymptotic convergence

Theorem. Let $\{x^k\}$ be iterate sequence generated by above greedy CD method. Then,

$$f(x_k) - f^* \leq \frac{2dM\|x_0 - x^*\|_2^2}{k+4}, \quad k \geq 0.$$

- Looks like gradient-descent $O(1/k)$ bound for C_L^1 cvx
- Notice factor of d in the numerator!
- But this method is impractical
- At each step, it requires access to **full gradient**
- Might as well use ordinary gradient methods!

CD – non-asymptotic convergence

Theorem. Let $\{x^k\}$ be iterate sequence generated by above greedy CD method. Then,

$$f(x_k) - f^* \le \frac{2dM\|x_0 - x^*\|_2^2}{k+4}, \quad k \ge 0.$$

- Looks like gradient-descent $O(1/k)$ bound for C_L^1 cvx
- Notice factor of d in the numerator!
- But this method is impractical
- At each step, it requires access to **full gradient**
- Might as well use ordinary gradient methods!
- Also, if $f \in C_L^1$, it can easily happen that $M \ge L$

CD – non-asymptotic convergence

Theorem. Let $\{x^k\}$ be iterate sequence generated by above greedy CD method. Then,

$$f(x_k) - f^* \leq \frac{2dM\|x_0 - x^*\|_2^2}{k+4}, \quad k \geq 0.$$

- Looks like gradient-descent $O(1/k)$ bound for C_L^1 cvx
- Notice factor of d in the numerator!
- But this method is impractical
- At each step, it requires access to **full gradient**
- Might as well use ordinary gradient methods!
- Also, if $f \in C_L^1$, it can easily happen that $M \geq L$
- So above rate is in general, worse than gradient methods

BCD – Notation

- **Decomposition:** $E = [E_1, \ldots, E_n]$ into n **blocks**
- Corresponding decomposition of x is

$$(\underbrace{E_1^T \boldsymbol{x}}_{N_1+}, \underbrace{E_2^T \boldsymbol{x}}_{N_2+}, \ldots, \underbrace{E_n^T \boldsymbol{x}}_{+N_n=N}) = (x^{(1)}, x^{(2)}, \ldots, x^{(n)})$$

- **Observation:**

$$E_i^T E_j = \begin{cases} I_{N_i} & i = j \\ 0_{N_i, N_j} & i \neq j. \end{cases}$$

- So the E_is define our partitioning of the coordinates
- Just fancier notation for a random partition of coordinates
- Now with this notation ...

BCD – formal setup

$$\boxed{\min\ f(\boldsymbol{x}) \text{ where } \boldsymbol{x} \in \mathbb{R}^d}$$

BCD – formal setup

$$\boxed{\min\ f(\boldsymbol{x}) \text{ where } \boldsymbol{x} \in \mathbb{R}^d}$$

Assume gradient of block i is Lipschitz continuous**

BCD – formal setup

$$\min\ f(\boldsymbol{x}) \text{ where } \boldsymbol{x} \in \mathbb{R}^d$$

Assume gradient of block i is Lipschitz continuous**

$$\|\nabla_i f(\boldsymbol{x} + E_i h) - \nabla_i f(\boldsymbol{x})\|_* \leq L_i \|h\|$$

Block gradient $\nabla_i f(\boldsymbol{x})$ is projection of full grad: $E_i^T \nabla f(\boldsymbol{x})$

BCD – formal setup

$$\boxed{\min\ f(\boldsymbol{x}) \text{ where } \boldsymbol{x} \in \mathbb{R}^d}$$

Assume gradient of block i is Lipschitz continuous**

$$\|\nabla_i f(\boldsymbol{x} + E_i h) - \nabla_i f(\boldsymbol{x})\|_* \leq L_i \|h\|$$

Block gradient $\nabla_i f(\boldsymbol{x})$ is projection of full grad: $E_i^T \nabla f(\boldsymbol{x})$
** — each block can use its own norm

BCD – formal setup

$$\min\ f(\boldsymbol{x}) \text{ where } \boldsymbol{x} \in \mathbb{R}^d$$

Assume gradient of block i is Lipschitz continuous**

$$\|\nabla_i f(\boldsymbol{x} + E_i h) - \nabla_i f(\boldsymbol{x})\|_* \leq L_i \|h\|$$

Block gradient $\nabla_i f(\boldsymbol{x})$ is projection of full grad: $E_i^T \nabla f(\boldsymbol{x})$
** — each block can use its own norm

Block Coordinate "Gradient" Descent

BCD – formal setup

$$\boxed{\min\ f(\boldsymbol{x}) \text{ where } \boldsymbol{x} \in \mathbb{R}^d}$$

Assume gradient of block i is Lipschitz continuous**

$$\|\nabla_i f(\boldsymbol{x} + E_i h) - \nabla_i f(\boldsymbol{x})\|_* \le L_i \|h\|$$

Block gradient $\nabla_i f(\boldsymbol{x})$ is projection of full grad: $E_i^T \nabla f(\boldsymbol{x})$
** — each block can use its own norm

Block Coordinate "Gradient" Descent

- Using the descent lemma, we have blockwise upper bounds

$$f(\boldsymbol{x} + E_i h) \le f(\boldsymbol{x}) + \langle \nabla_i f(\boldsymbol{x}), h\rangle + \tfrac{L_i}{2}\|h\|^2, \quad \text{for } i = 1, \ldots, d.$$

BCD – formal setup

$$\boxed{\min\ f(\boldsymbol{x}) \text{ where } \boldsymbol{x} \in \mathbb{R}^d}$$

Assume gradient of block i is Lipschitz continuous**

$$\|\nabla_i f(\boldsymbol{x} + E_i h) - \nabla_i f(\boldsymbol{x})\|_* \le L_i \|h\|$$

Block gradient $\nabla_i f(\boldsymbol{x})$ is projection of full grad: $E_i^T \nabla f(\boldsymbol{x})$
** — each block can use its own norm

Block Coordinate "Gradient" Descent

- Using the descent lemma, we have blockwise upper bounds

$$f(\boldsymbol{x} + E_i h) \le f(\boldsymbol{x}) + \langle \nabla_i f(\boldsymbol{x}), h \rangle + \tfrac{L_i}{2}\|h\|^2, \quad \text{for } i = 1, \ldots, d.$$

- At each step, minimize these upper bounds!

Randomized BCD

- For $k \geq 0$ (no init. of $\boldsymbol{x}$ necessary)

Randomized BCD

- For $k \geq 0$ (no init. of $\boldsymbol{x}$ necessary)
- Pick a block i from $[d]$ with probability $p_i > 0$

Randomized BCD

- For $k \geq 0$ (no init. of $\boldsymbol{x}$ necessary)
- Pick a block i from $[d]$ with probability $p_i > 0$
- Optimize upper bound (partial gradient step) for block i

$$h = \operatorname*{argmin}_h f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k),\, h \rangle + \tfrac{L_i}{2}\|h\|^2$$
$$h = -\tfrac{1}{L_i}\nabla_i f(\boldsymbol{x}_k)$$

Randomized BCD

- For $k \geq 0$ (no init. of $\boldsymbol{x}$ necessary)
- Pick a block i from $[d]$ with probability $p_i > 0$
- Optimize upper bound (partial gradient step) for block i

$$h = \operatorname*{argmin}_h f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k), h \rangle + \tfrac{L_i}{2}\|h\|^2$$
$$h = -\tfrac{1}{L_i}\nabla_i f(\boldsymbol{x}_k)$$

- Update the impacted coordinates of $\boldsymbol{x}$, formally

Randomized BCD

- For $k \geq 0$ (no init. of $\boldsymbol{x}$ necessary)
- Pick a block i from $[d]$ with probability $p_i > 0$
- Optimize upper bound (partial gradient step) for block i

$$h = \operatorname*{argmin}_h f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k), h \rangle + \tfrac{L_i}{2}\|h\|^2$$
$$h = -\tfrac{1}{L_i}\nabla_i f(\boldsymbol{x}_k)$$

- Update the impacted coordinates of $\boldsymbol{x}$, formally

$$\boldsymbol{x}_{k+1}^{(i)} \leftarrow \boldsymbol{x}_k^{(i)} + h$$
$$\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k - \tfrac{1}{L_i} E_i \nabla f(\boldsymbol{x}_k)$$

Randomized BCD

- For $k \geq 0$ (no init. of $\boldsymbol{x}$ necessary)
- Pick a block i from $[d]$ with probability $p_i > 0$
- Optimize upper bound (partial gradient step) for block i

$$h = \operatorname*{argmin}_h f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k), h \rangle + \tfrac{L_i}{2}\|h\|^2$$
$$h = -\tfrac{1}{L_i}\nabla_i f(\boldsymbol{x}_k)$$

- Update the impacted coordinates of $\boldsymbol{x}$, formally

$$\boldsymbol{x}_{k+1}^{(i)} \leftarrow \boldsymbol{x}_k^{(i)} + h$$
$$\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k - \tfrac{1}{L_i} E_i \nabla f(\boldsymbol{x}_k)$$

Notice: Original BCD had: $x_k^{(i)} = \operatorname{argmin}_h f(\ldots, \underbrace{h}_{\text{block } i}, \ldots)$

Randomized BCD

- For $k \geq 0$ (no init. of $\boldsymbol{x}$ necessary)
- Pick a block i from $[d]$ with probability $p_i > 0$
- Optimize upper bound (partial gradient step) for block i

$$h = \underset{h}{\operatorname{argmin}} f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k), h \rangle + \tfrac{L_i}{2}\|h\|^2$$

$$h = -\tfrac{1}{L_i} \nabla_i f(\boldsymbol{x}_k)$$

- Update the impacted coordinates of $\boldsymbol{x}$, formally

$$\boldsymbol{x}_{k+1}^{(i)} \leftarrow \boldsymbol{x}_k^{(i)} + h$$

$$\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k - \tfrac{1}{L_i} E_i \nabla f(\boldsymbol{x}_k)$$

Notice: Original BCD had: $x_k^{(i)} = \operatorname{argmin}_h f(\ldots, \underbrace{h}_{\text{block } i}, \ldots)$

We'll call this BCM (**Block Coordinate Minimization**)

Exercise: proximal extension

$$\min f(\boldsymbol{x}) + r(\boldsymbol{x})$$

- If **block separable** $r(\boldsymbol{x}) := \sum_{i=1}^{n} r_i(x^{(i)})$

$$x_k^{(i)} = \underset{h}{\operatorname{argmin}} f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k), h \rangle + \tfrac{L_i}{2}\|h\|^2 + r_i(E_i^T \boldsymbol{x}_k + h)$$

$$x_k^{(i)} = \operatorname{prox}_{r_i}(\cdots)$$

Exercise: Fill in the dots

$$h = \operatorname{prox}_{(1/L)r_i}\left(E_i^T \boldsymbol{x}_k - \tfrac{1}{L_i}\nabla_i f(\boldsymbol{x}_k)\right)$$

Randomized BCD – analysis

$$\boxed{h \leftarrow \operatorname{argmin}_h f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k), h \rangle + \tfrac{L_i}{2}\|h\|^2}$$

Randomized BCD – analysis

$$\boxed{h \leftarrow \operatorname{argmin}_h\ f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k),\, h \rangle + \tfrac{L_i}{2}\|h\|^2}$$

Descent:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{x}_k + E_i h \\ f(\boldsymbol{x}_{k+1}) &\leq f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k),\, h \rangle + \tfrac{L_i}{2}\|h\|^2 \end{aligned}$$

Randomized BCD – analysis

$$\boxed{h \leftarrow \operatorname{argmin}_h f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k), h\rangle + \tfrac{L_i}{2}\|h\|^2}$$

Descent:

$$\begin{aligned}
\boldsymbol{x}_{k+1} &= \boldsymbol{x}_k + E_i h \\
f(\boldsymbol{x}_{k+1}) &\leq f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k), h\rangle + \tfrac{L_i}{2}\|h\|^2 \\
\boldsymbol{x}_{k+1} &= \boldsymbol{x}_k - \tfrac{1}{L_i} E_i \nabla_i f(\boldsymbol{x}_k)
\end{aligned}$$

Randomized BCD – analysis

$$\boxed{h \leftarrow \operatorname{argmin}_h\ f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k),\ h \rangle + \tfrac{L_i}{2}\|h\|^2}$$

Descent:

$$\begin{aligned}
\boldsymbol{x}_{k+1} &= \boldsymbol{x}_k + E_i h \\
f(\boldsymbol{x}_{k+1}) &\leq f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k),\ h \rangle + \tfrac{L_i}{2}\|h\|^2 \\
\boldsymbol{x}_{k+1} &= \boldsymbol{x}_k - \tfrac{1}{L_i} E_i \nabla_i f(\boldsymbol{x}_k) \\
f(\boldsymbol{x}_{k+1}) &\leq f(\boldsymbol{x}_k) - \tfrac{1}{L_i}\|\nabla_i f(\boldsymbol{x}_k)\|^2 + \tfrac{L_i}{2}\left\| -\tfrac{1}{L_i}\nabla_i f(\boldsymbol{x}_k) \right\|^2
\end{aligned}$$

Randomized BCD – analysis

$$\boxed{h \leftarrow \operatorname{argmin}_h f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k), h \rangle + \tfrac{L_i}{2}\|h\|^2}$$

Descent:

$$\begin{aligned}
\boldsymbol{x}_{k+1} &= \boldsymbol{x}_k + E_i h \\
f(\boldsymbol{x}_{k+1}) &\leq f(\boldsymbol{x}_k) + \langle \nabla_i f(\boldsymbol{x}_k), h \rangle + \tfrac{L_i}{2}\|h\|^2 \\
\boldsymbol{x}_{k+1} &= \boldsymbol{x}_k - \tfrac{1}{L_i} E_i \nabla_i f(\boldsymbol{x}_k) \\
f(\boldsymbol{x}_{k+1}) &\leq f(\boldsymbol{x}_k) - \tfrac{1}{L_i}\|\nabla_i f(\boldsymbol{x}_k)\|^2 + \tfrac{L_i}{2}\left\| -\tfrac{1}{L_i}\nabla_i f(\boldsymbol{x}_k)\right\|^2 \\
f(\boldsymbol{x}_{k+1}) &\leq f(\boldsymbol{x}_k) - \tfrac{1}{2L_i}\|\nabla_i f(\boldsymbol{x}_k)\|^2.
\end{aligned}$$

$$\boxed{f(\boldsymbol{x}_k) - f(\boldsymbol{x}_{k+1}) \geq \tfrac{1}{2L_i}\|\nabla_i f(\boldsymbol{x}_k)\|^2}$$

Randomized BCD – analysis

Expected descent:

$$f(\boldsymbol{x}_k) - \mathbb{E}[f(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)] \quad = \quad \sum_{i=1}^{d} p_i \big(f(\boldsymbol{x}_k) - f(\boldsymbol{x}_k - \tfrac{1}{L_i} E_i \nabla_i f(\boldsymbol{x}_k))\big)$$

Randomized BCD – analysis

Expected descent:

$$
\begin{aligned}
f(\boldsymbol{x}_k) - \mathbb{E}[f(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)] &= \sum_{i=1}^{d} p_i \Big(f(\boldsymbol{x}_k) - f(\boldsymbol{x}_k - \tfrac{1}{L_i} E_i \nabla_i f(\boldsymbol{x}_k))\Big) \\
&\geq \sum_{i=1}^{d} \tfrac{p_i}{2L_i} \|\nabla_i f(\boldsymbol{x}_k)\|^2
\end{aligned}
$$

Randomized BCD – analysis

Expected descent:

$$
\begin{aligned}
f(\boldsymbol{x}_k) - \mathbb{E}[f(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)] &= \sum_{i=1}^{d} p_i\Big(f(\boldsymbol{x}_k) - f(\boldsymbol{x}_k - \tfrac{1}{L_i} E_i \nabla_i f(\boldsymbol{x}_k))\Big) \\
&\geq \sum_{i=1}^{d} \tfrac{p_i}{2L_i} \|\nabla_i f(\boldsymbol{x}_k)\|^2 \\
&= \tfrac{1}{2}\|\nabla f(\boldsymbol{x}_k)\|_W^2 \quad \text{(suitable } W\text{)}.
\end{aligned}
$$

Randomized BCD – analysis

Expected descent:

$$
\begin{aligned}
f(\boldsymbol{x}_k) - \mathbb{E}[f(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)] &= \sum_{i=1}^{d} p_i\big(f(\boldsymbol{x}_k) - f(\boldsymbol{x}_k - \tfrac{1}{L_i}E_i\nabla_i f(\boldsymbol{x}_k))\big) \\
&\geq \sum_{i=1}^{d} \tfrac{p_i}{2L_i}\|\nabla_i f(\boldsymbol{x}_k)\|^2 \\
&= \tfrac{1}{2}\|\nabla f(\boldsymbol{x}_k)\|_W^2 \quad \text{(suitable } W\text{)}.
\end{aligned}
$$

Exercise: What's expected descent with uniform probabilities?

Randomized BCD – analysis

Expected descent:

$$
\begin{aligned}
f(\boldsymbol{x}_k) - \mathbb{E}[f(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)] &= \sum_{i=1}^{d} p_i \big(f(\boldsymbol{x}_k) - f(\boldsymbol{x}_k - \tfrac{1}{L_i} E_i \nabla_i f(\boldsymbol{x}_k))\big) \\
&\geq \sum_{i=1}^{d} \tfrac{p_i}{2L_i} \|\nabla_i f(\boldsymbol{x}_k)\|^2 \\
&= \tfrac{1}{2} \|\nabla f(\boldsymbol{x}_k)\|_W^2 \quad \text{(suitable } W\text{)}.
\end{aligned}
$$

Exercise: What's expected descent with uniform probabilities?

Descent plus some more (hard) work yields

$$
O\Big(\frac{d}{\epsilon} \sum\nolimits_i L_i \|x_0^{(i)} - x_*^{(i)}\|^2\Big)
$$

as the iteration complexity of obtaining $\mathbb{E}[f(\boldsymbol{x}_k)] - f^* \leq \epsilon$

BCD – Exercise

- Recall Lasso problem: $\min \frac{1}{2}\|Ax - b\|^2 + \lambda\|x\|_1$
- Here $x \in \mathbb{R}^N$
- Make $n = N$ blocks
- Show what the Randomized BCD iterations look like
- Notice, 1D prox operations for $\lambda|\cdot|$ arise
- Try to implement it as efficiently as you can (i.e., do not copy or update vectors / coordinates than necessary)

Connections

CD – exercise

$$\min \quad \frac{1}{n}\sum_{i=1}^{n} f_i(x^T a_i) + \frac{\lambda}{2}\|x\|^2.$$

Dual problem

$$\max_{\alpha} \quad \frac{1}{n}\sum_{i=1}^{n} -f_i^*(-\alpha_i) - \frac{\lambda}{2}\left\|\frac{1}{\lambda n}\sum_{i=1}^{n}\alpha_i a_i\right\|^2$$

Exercise: Study the SDCA algorithm and derive a connection between it and SAG/SAGA family of algorithms.

S. Shalev-Shwartz, T. Zhang. *Stochastic Dual Coordinate Ascent Methods for Regularized Loss Minimization*. JMLR (2013).

Other connections

Explore: Block-Coordinate Frank-Wolfe algorithm.

$$\min_x \quad f(x), \quad \text{s.t. } x \in \prod_i \mathcal{X}_i$$

Explore: Doubly stochastic methods

$$\min \quad f(x) = \sum\nolimits_i f_i(x_1, \ldots, x_d)$$

Being jointly stochastic over f_i as well as coordinates.

Explore: CD with constraints (linear and nonlinear constraints)