CS6301: Optimization in Machine Learning

Lecture 14: Coordinate Descent Family

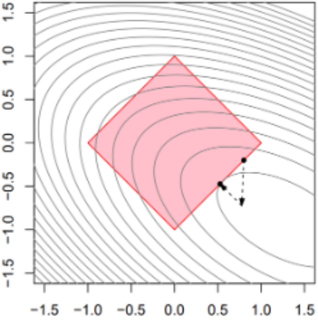
Rishabh Iyer

Department of Computer Science
University of Texas, Dallas
https://sites.google.com/view/cs-6301-optml/home

March 9th, 2020

Recall: Projected Gradient Descent

- Consider the Problem of Constrained Convex Minimization: $\min_{x \in \mathcal{C}} f(x)$
- A simple modification of the gradient descent procedure is:
  1. At every iteration t: (Gradient Step): Compute $y_{t+1} = x_t - \alpha \nabla f(x_t)$
  2. (Projection step) $x_{t+1} = P_{\mathcal{C}}(y_{t+1}$
- Key here is the Projection step. Define $P_{\mathcal{C}}(x) = \text{argmin}_{y \in \mathcal{C}} \frac{1}{2}||x - y||^2$



Recall: Conditional Gradient Descent Algorithm

$$
\begin{aligned}
&\textbf{Input} : \text{initial guess } \boldsymbol{x}_0, \text{ tolerance } \delta > 0 && \\
&\textbf{For } t = 0, 1, \ldots \textbf{ do} && (2)\\
&\quad \boldsymbol{s}_t \in \arg\max_{\boldsymbol{s} \in \mathcal{D}} \langle -\nabla f(\boldsymbol{x}_t), \boldsymbol{s} \rangle && (3)\\
&\quad \boldsymbol{d}_t = \boldsymbol{s}_t - \boldsymbol{x}_t && (4)\\
&\quad g_t = -\langle \nabla f(\boldsymbol{x}_t), \boldsymbol{d}_t \rangle && (5)\\
&\quad \textbf{If } g_t < \delta : && (6)\\
&\qquad // \text{ exit if gap is below tolerance} && \\
&\qquad \textbf{return } \boldsymbol{x}_t && (7)\\
&\quad \textbf{Variant 1} : \text{set step size as} && \\
&\qquad \gamma_t = \min\left\{ \frac{g_t}{L\|\boldsymbol{d}_t\|^2}, 1 \right\} && (8)\\
&\quad \textbf{Variant 2} : \text{set step size by line search} && \\
&\qquad \gamma_t = \arg\min_{\gamma \in [0,1]} f(\boldsymbol{x}_t + \gamma \boldsymbol{d}_t) && (9)\\
&\quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \gamma_t \boldsymbol{d}_t \ . && (10)\\
&\textbf{end For loop} && (11)\\
&\textbf{return } \boldsymbol{x}_t && (12)
\end{aligned}
$$

Projected Gradient vs Conditional Gradient

- Projected Gradient Descent requires optimizing a quadratic function over the constraints

Projected Gradient vs Conditional Gradient

- Projected Gradient Descent requires optimizing a quadratic function over the constraints
- While it is easy for a few constraints, several common constraints do not admit easy projection operators

Projected Gradient vs Conditional Gradient

- Projected Gradient Descent requires optimizing a quadratic function over the constraints
- While it is easy for a few constraints, several common constraints do not admit easy projection operators
- Conditional Gradient Descent requires solving a linear program over constraints

Projected Gradient vs Conditional Gradient

- Projected Gradient Descent requires optimizing a quadratic function over the constraints
- While it is easy for a few constraints, several common constraints do not admit easy projection operators
- Conditional Gradient Descent requires solving a linear program over constraints
- This is easy for a large class of constraints (e.g. combinatorial constraints, polyhedral constraints etc.)

Projected Gradient vs Conditional Gradient

- Projected Gradient Descent requires optimizing a quadratic function over the constraints
- While it is easy for a few constraints, several common constraints do not admit easy projection operators
- Conditional Gradient Descent requires solving a linear program over constraints
- This is easy for a large class of constraints (e.g. combinatorial constraints, polyhedral constraints etc.)
- In contrast, conditional gradient can be slower in terms of convergence

Projected Gradient vs Conditional Gradient

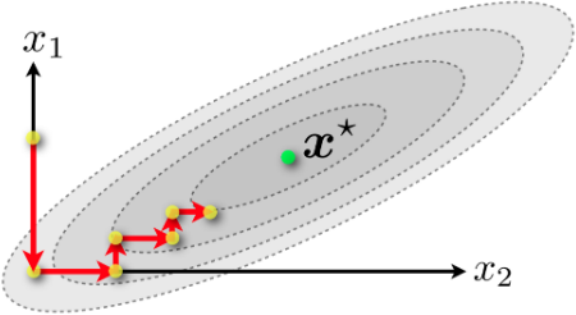
- Projected Gradient Descent requires optimizing a quadratic function over the constraints
- While it is easy for a few constraints, several common constraints do not admit easy projection operators
- Conditional Gradient Descent requires solving a linear program over constraints
- This is easy for a large class of constraints (e.g. combinatorial constraints, polyhedral constraints etc.)
- In contrast, conditional gradient can be slower in terms of convergence
- Takeaways: If projection is easy, use projected gradient descent. Else, conditional gradient is the go to algorithm!

Outline of this Lecture

- Coordinate Descent: Basic Idea
- Optimality w.r.t co-ordinate descent
- Two variants of Coordinate Descent
- Convergence of coordinate descent algorithms.

Coordinate Descent

Goal: Find $\mathbf{x}^\star \in \mathbb{R}^d$ minimizing $f(\mathbf{x})$.



Idea: Update one coordinate at a time, while keeping others fixed.

Coordinate Descent

- Modify only one coordinate per step:

Coordinate Descent

- Modify only one coordinate per step:
  1. Select $i_k \in [d]$

Coordinate Descent

- Modify only one coordinate per step:
  1. Select $i_k \in [d]$
  2. $x_{k+1} = x_k + \gamma e_{i_k}$

Coordinate Descent

- Modify only one coordinate per step:
  1. Select $i_k \in [d]$
  2. $x_{k+1} = x_k + \gamma e_{i_k}$
- γ is the step size of coordinate descent

Coordinate Descent

- Modify only one coordinate per step:
  1. Select $i_k \in [d]$
  2. $x_{k+1} = x_k + \gamma e_{i_k}$
- γ is the step size of coordinate descent
- Two variants of coordinate descent:

Coordinate Descent

- Modify only one coordinate per step:
  1. Select $i_k \in [d]$
  2. $x_{k+1} = x_k + \gamma e_{i_k}$
- γ is the step size of coordinate descent
- Two variants of coordinate descent:
- Gradient based step size:

$$x_{k+1} = x_k - \frac{1}{L} \nabla_{i_k} f(x_k) e_{i_k}$$

Coordinate Descent

- Modify only one coordinate per step:
  1. Select $i_k \in [d]$
  2. $x_{k+1} = x_k + \gamma e_{i_k}$
- γ is the step size of coordinate descent
- Two variants of coordinate descent:
- Gradient based step size:

$$x_{k+1} = x_k - \frac{1}{L} \nabla_{i_k} f(x_k) e_{i_k}$$

- Exact coordinate minimization: solve the single variable minimization

$$\text{argmin}_{\gamma \in R} f(x_k + \gamma e_{i_k})$$

in closed form.

Coordinate Descent

- Two variants of coordinate descent:

Coordinate Descent

- Two variants of coordinate descent:
- Gradient based step size:

$$x_{k+1} = x_k - \frac{1}{L}\nabla_{i_k} f(x_k) e_{i_k}$$

Coordinate Descent

- Two variants of coordinate descent:
- Gradient based step size:

$$x_{k+1} = x_k - \frac{1}{L}\nabla_{i_k} f(x_k) e_{i_k}$$

- Exact coordinate minimization: solve the single variable minimization

$$\text{argmin}_{\gamma \in R} f(x_k + \gamma e_{i_k})$$

in closed form.

Coordinate Descent

- Two variants of coordinate descent:
- Gradient based step size:

$$x_{k+1} = x_k - \frac{1}{L}\nabla_{i_k} f(x_k) e_{i_k}$$

- Exact coordinate minimization: solve the single variable minimization

$$\text{argmin}_{\gamma \in R} f(x_k + \gamma e_{i_k})$$

  in closed form.
- Also how do we select the coordinate i_k?

Coordinate Descent

- Two variants of coordinate descent:
- Gradient based step size:

$$x_{k+1} = x_k - \frac{1}{L}\nabla_{i_k} f(x_k) e_{i_k}$$

- Exact coordinate minimization: solve the single variable minimization

$$\text{argmin}_{\gamma \in R} f(x_k + \gamma e_{i_k})$$

  in closed form.
- Also how do we select the coordinate i_k?
- Two strategies: One is to pick it up randomly (called randomized coordinate descent) and second it to pick it up greedily (greedy coordinate descent).

Randomized Coordinate Descent

$$\text{select } i_t \in [d] \text{ uniformly at random}$$
$$\mathbf{x}_{t+1} := \mathbf{x}_t - \tfrac{1}{L}\nabla_{i_t} f(\mathbf{x}_t)\, \mathbf{e}_{i_t}$$

- Faster convergence than gradient descent
  (if coordinate step is significantly cheaper than full gradient step)

Randomized Coordinate Descent: Convergence Analysis

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

Randomized Coordinate Descent: Convergence Analysis

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- This is equivalent to coordinate wise Lipschitz gradient:

$$|\nabla_i f(x + \gamma e_i) - \nabla_i f(x)| \leq L_i |\gamma|$$

Randomized Coordinate Descent: Convergence Analysis

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- This is equivalent to coordinate wise Lipschitz gradient:

$$|\nabla_i f(x + \gamma e_i) - \nabla_i f(x)| \leq L_i |\gamma|$$

- Define $L = \max_i L_i$ as the Maximum coordinate wise Lipschitz smoothness

Randomized Coordinate Descent: Convergence Analysis

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- This is equivalent to coordinate wise Lipschitz gradient:

$$|\nabla_i f(x + \gamma e_i) - \nabla_i f(x)| \leq L_i |\gamma|$$

- Define $L = \max_i L_i$ as the Maximum coordinate wise Lipschitz smoothness
- Similar to previous analysis, lets assume $||x_0 - x^*||^2 \leq R^2$.

Randomized Coordinate Descent: Convergence Result

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

Randomized Coordinate Descent: Convergence Result

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Define $L = \max_i L_i$ as the Maximum coordinate wise Lipschitz smoothness. Also, assume $||x_0 - x^*||^2 \leq R^2$.

Randomized Coordinate Descent: Convergence Result

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x+\gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Define $L = \max_i L_i$ as the Maximum coordinate wise Lipschitz smoothness. Also, assume $||x_0 - x^*||^2 \leq R^2$.
- **Theorem:** For a coordinate wise smooth convex function, we have:

$$E(f(x_k)) - f_* \leq \frac{2dLR_0^2}{k}$$

Randomized Coordinate Descent: Convergence Result

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Define $L = \max_i L_i$ as the Maximum coordinate wise Lipschitz smoothness. Also, assume $||x_0 - x^*||^2 \leq R^2$.
- **Theorem:** For a coordinate wise smooth convex function, we have:

$$E(f(x_k)) - f_* \leq \frac{2dLR_0^2}{k}$$

- Similar to Gradient Descent, this is $O(1/\epsilon)$ convergence!

Randomized Coordinate Descent: Convergence Result

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Define $L = \max_i L_i$ as the Maximum coordinate wise Lipschitz smoothness. Also, assume $||x_0 - x^*||^2 \leq R^2$.
- **Theorem:** For a coordinate wise smooth convex function, we have:

$$E(f(x_k)) - f_* \leq \frac{2dLR_0^2}{k}$$

- Similar to Gradient Descent, this is $O(1/\epsilon)$ convergence!
- However, we just need to update a single dimension at a time, and hence the per iteration cost of CD can be $O(d)$ cheaper compared to GD!

Randomized Coordinate Descent: Convergence Result

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

Randomized Coordinate Descent: Convergence Result

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Define $L = \max_i L_i$ as the Maximum coordinate wise Lipschitz smoothness. Also, assume $||x_0 - x^*||^2 \leq R^2$.

Randomized Coordinate Descent: Convergence Result

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Define $L = \max_i L_i$ as the Maximum coordinate wise Lipschitz smoothness. Also, assume $||x_0 - x^*||^2 \leq R^2$.
- If additionally, f is μ-strongly convex? (Recall strong convexity implies $f(y) \geq f(x) + \nabla f(x)^T(y - x) + \frac{\mu}{2}||y - x||^2$, for all x, y)

Randomized Coordinate Descent: Convergence Result

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Define $L = \max_i L_i$ as the Maximum coordinate wise Lipschitz smoothness. Also, assume $||x_0 - x^*||^2 \leq R^2$.
- If additionally, f is μ-strongly convex? (Recall strong convexity implies $f(y) \geq f(x) + \nabla f(x)^T(y - x) + \frac{\mu}{2}||y - x||^2$, for all x, y)
- **Theorem:** If f is coordinate wise Lipschitz Smooth + Strongly Convex, we have:

$$E(f(x_k)) - f_* \leq (1 - \frac{\mu}{dL})^k (f(x_0) - f_*)$$

Randomized Coordinate Descent: Improvement at Every iteration!

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

Randomized Coordinate Descent: Improvement at Every iteration!

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Define $L = \max_i L_i$ as the Maximum coordinate wise Lipschitz smoothness. Also, assume $||x_0 - x^*||^2 \leq R^2$.

Randomized Coordinate Descent: Improvement at Every iteration!

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Define $L = \max_i L_i$ as the Maximum coordinate wise Lipschitz smoothness. Also, assume $||x_0 - x^*||^2 \leq R^2$.
- It holds that:

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

Randomized Coordinate Descent: Improvement at Every iteration!

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x+\gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Define $L = \max_i L_i$ as the Maximum coordinate wise Lipschitz smoothness. Also, assume $||x_0 - x^*||^2 \leq R^2$.
- It holds that:

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

- This means that similar to GD, CD improves the objective value at every iteration!

Importance Sampling

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

Importance Sampling

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Instead of random selection, can we be slightly smarter?

Importance Sampling

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x+\gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Instead of random selection, can we be slightly smarter?
- Sample i_k with probability: $P[i_k = i] = \frac{L_i}{\sum_i L_i}$, and use step size $1/L_{i_k}$, we can get a slightly better rate of convergence

Importance Sampling

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Instead of random selection, can we be slightly smarter?
- Sample i_k with probability: $P[i_k = i] = \frac{L_i}{\sum_i L_i}$, and use step size $1/L_{i_k}$, we can get a slightly better rate of convergence
  1. Smooth Functions: $\frac{2d\bar{L}R_0^2}{k}$

Importance Sampling

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Instead of random selection, can we be slightly smarter?
- Sample i_k with probability: $P[i_k = i] = \frac{L_i}{\sum_i L_i}$, and use step size $1/L_{i_k}$, we can get a slightly better rate of convergence
  1. Smooth Functions: $\frac{2d\bar{L}R_0^2}{k}$
  2. Smooth + Strongly Convex: $(1 - \frac{\mu}{d\bar{L}})^k (f(x_0) - f_*)$

Importance Sampling

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Instead of random selection, can we be slightly smarter?
- Sample i_k with probability: $P[i_k = i] = \frac{L_i}{\sum_i L_i}$, and use step size $1/L_{i_k}$, we can get a slightly better rate of convergence
  1. Smooth Functions: $\frac{2d\bar{L}R_0^2}{k}$
  2. Smooth + Strongly Convex: $(1 - \frac{\mu}{d\bar{L}})^k(f(x_0) - f_*)$
- In the above, $\bar{L} = \sum_i L_i/d$

Importance Sampling

- Key ingredient: **Coordinate wise Smoothness**:

$$f(x + \gamma e_i) \leq f(x) + \gamma \nabla_i f(x) + \frac{L_i}{2}\gamma^2, \forall x \in \mathbb{R}^d, \forall \gamma \in \mathbb{R}$$

- Instead of random selection, can we be slightly smarter?
- Sample i_k with probability: $P[i_k = i] = \frac{L_i}{\sum_i L_i}$, and use step size $1/L_{i_k}$, we can get a slightly better rate of convergence
  1. Smooth Functions: $\frac{2d\bar{L}R_0^2}{k}$
  2. Smooth + Strongly Convex: $(1 - \frac{\mu}{d\bar{L}})^k(f(x_0) - f_*)$
- In the above, $\bar{L} = \sum_i L_i/d$
- Note that $\bar{L} \leq L$ and can in fact be significantly smaller!

Steepest (or Greedy) Coordinate Descent

- Instead of random or importance sampling based selection, select greedily!

Steepest (or Greedy) Coordinate Descent

- Instead of random or importance sampling based selection, select greedily!
- Select:

$$i_k = \text{argmax}_{i \in [d]} [\nabla_i f(x_k)]$$

Steepest (or Greedy) Coordinate Descent

- Instead of random or importance sampling based selection, select greedily!
- Select:

$$i_k = \text{argmax}_{i \in [d]} [\nabla_i f(x_k)]$$

- Note, this strategy is deterministic instead of random.

Steepest (or Greedy) Coordinate Descent

- Instead of random or importance sampling based selection, select greedily!
- Select:

$$i_k = \text{argmax}_{i \in [d]}[\nabla_i f(x_k)]$$

- Note, this strategy is deterministic instead of random.
- Vanilla analysis gives the same worst case Convergence rate as the randomized case.

Steepest (or Greedy) Coordinate Descent

- Instead of random or importance sampling based selection, select greedily!
- Select:

$$i_k = \text{argmax}_{i \in [d]} [\nabla_i f(x_k)]$$

- Note, this strategy is deterministic instead of random.
- Vanilla analysis gives the same worst case Convergence rate as the randomized case.
- However, if we assume f is strongly convex w.r.t the l_1 norm with parameter $\mu_1 > 0$, we get improved rates of:

Steepest (or Greedy) Coordinate Descent

- Instead of random or importance sampling based selection, select greedily!
- Select:

$$i_k = \text{argmax}_{i \in [d]} [\nabla_i f(x_k)]$$

- Note, this strategy is deterministic instead of random.
- Vanilla analysis gives the same worst case Convergence rate as the randomized case.
- However, if we assume f is strongly convex w.r.t the l_1 norm with parameter $\mu_1 > 0$, we get improved rates of:
  1. Smooth Functions: $\frac{2LR_0^2}{k}$

Steepest (or Greedy) Coordinate Descent

- Instead of random or importance sampling based selection, select greedily!
- Select:

$$i_k = \text{argmax}_{i \in [d]} [\nabla_i f(x_k)]$$

- Note, this strategy is deterministic instead of random.
- Vanilla analysis gives the same worst case Convergence rate as the randomized case.
- However, if we assume f is strongly convex w.r.t the l_1 norm with parameter $\mu_1 > 0$, we get improved rates of:
  1. Smooth Functions: $\frac{2LR_0^2}{k}$
  2. Smooth + Strongly Convex: $(1 - \frac{\sigma}{L})^k (f(x_0) - f_*)$

Exact Coordinate Minimization

- So far, we went over coordinate descent in a way akin to gradient descent, i.e. taking a coordinate step in each coordinate.
- Next, consider exact minimization in each coordinate.

$$
\begin{aligned}
x_1^{(k)} &\in \operatorname*{argmin}_{x_1} f\left(x_1, x_2^{(k-1)}, x_3^{(k-1)}, \dots x_n^{(k-1)}\right) \\
x_2^{(k)} &\in \operatorname*{argmin}_{x_2} f\left(x_1^{(k)}, x_2, x_3^{(k-1)}, \dots x_n^{(k-1)}\right) \\
x_3^{(k)} &\in \operatorname*{argmin}_{x_2} f\left(x_1^{(k)}, x_2^{(k)}, x_3, \dots x_n^{(k-1)}\right) \\
&\dots \\
x_n^{(k)} &\in \operatorname*{argmin}_{x_2} f\left(x_1^{(k)}, x_2^{(k)}, x_3^{(k)}, \dots x_n\right)
\end{aligned}
$$

for $k = 1, 2, 3, \dots$

Note: after we solve for $x_i^{(k)}$, we use its new value from then on!

Exact Coordinate Minimization: Linear Regression

Consider linear regression

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{2}\|y - X\beta\|_2^2$$

where $y \in \mathbb{R}^n$, and $X \in \mathbb{R}^{n \times p}$ with columns $X_1, \ldots X_p$

Minimizing over β_i, with all β_j, $j \neq i$ fixed:

$$0 = \nabla_i f(\beta) = X_i^T(X\beta - y) = X_i^T(X_i\beta_i + X_{-i}\beta_{-i} - y)$$

i.e., we take

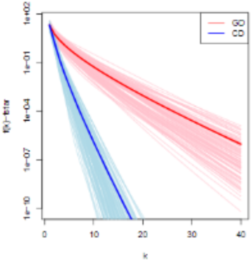
$$\beta_i = \frac{X_i^T(y - X_{-i}\beta_{-i})}{X_i^T X_i}$$

Coordinate descent repeats this update for $i = 1, 2, \ldots, p, 1, 2, \ldots$

LAS

Exact Coordinate Minimization: Linear Regression

Coordinate descent vs gradient descent for linear regression: 100 instances ($n = 100$, $p = 20$)



Is it fair to compare 1 cycle of coordinate descent to 1 iteration of gradient descent? Yes, if we're clever:

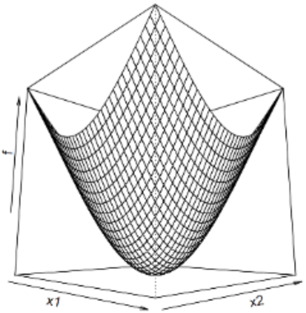
$$\beta_i \leftarrow \frac{X_i^T(y - X_{-i}\beta_{-i})}{X_i^T X_i} = \frac{X_i^T r}{\|X_i\|_2^2} + \beta_i$$

where $r = y - X\beta$. Therefore each coordinate update takes $O(n)$ operations — $O(n)$ to update r, and $O(n)$ to compute $X_i^T r$ — and one cycle requires $O(np)$ operations, just like gradient descent

Optimality Conditions with Coordinate Descent

- Coordinate Descent Family of Algorithms try to make progress every iteration!
- What is the stopping/optimality condition?
- One way to think of this is: If we are a point x such that it is minimum along each coordinate axis! In other words, no coordinate descent algorithm can make progress!
- Does this mean we are at a global minimum?
- Mathematically this means: Does a point x satisfying $f(x + \delta e_i) \geq f(x), \forall \delta, i \Rightarrow f(x) = \min_z f(z)$?

Optimality Conditions: Differentiable Functions

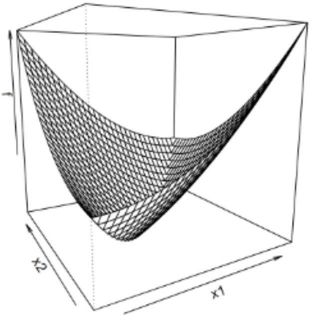


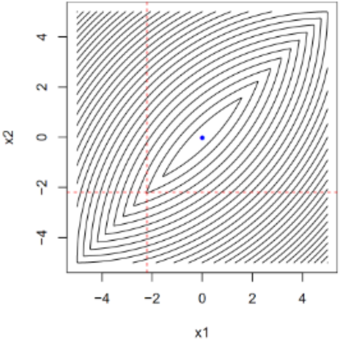
A: Yes! Proof:

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1}(x), \ldots \frac{\partial f}{\partial x_n}(x) \right) = 0$$

Q: Same question, but for f convex (not differentiable) ... ?

Optimality Conditions: Non Differentiable Functions

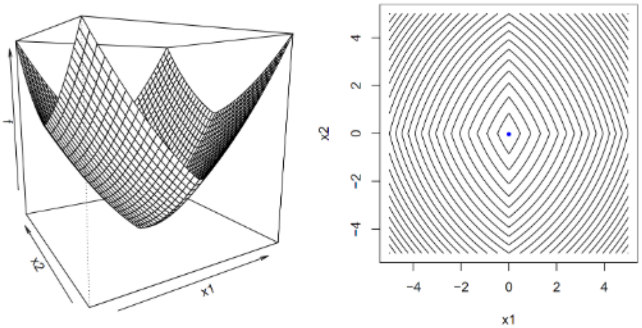




A: No! Look at the above counterexample

Q: Same question again, but now $f(x) = g(x) + \sum_{i=1}^{n} h_i(x_i)$, with g convex, differentiable and each h_i convex ... ? (Nonsmooth part here called separable)

Optimality Conditions: Seperable Functions



A: Yes! Proof: for any y,

$$f(y) - f(x) \geq \nabla g(x)^T (y - x) + \sum_{i=1}^{n} [h_i(y_i) - h_i(x_i)]$$

$$= \sum_{i=1}^{n} \underbrace{[\nabla_i g(x)(y_i - x_i) + h_i(y_i) - h_i(x_i)]}_{\geq 0} \geq 0$$

Coordinate Descent Summary

- Algorithms:

Coordinate Descent Summary

- Algorithms:
  1. Randomized Coordinate Descent

Coordinate Descent Summary

- Algorithms:
  1. Randomized Coordinate Descent
  2. Greedy (Steepest) Coordinate Descent

Coordinate Descent Summary

- Algorithms:
  1. Randomized Coordinate Descent
  2. Greedy (Steepest) Coordinate Descent
  3. Exact Coordinate Minimization

Coordinate Descent Summary

- Algorithms:
  1. Randomized Coordinate Descent
  2. Greedy (Steepest) Coordinate Descent
  3. Exact Coordinate Minimization
- Convergence Results:

Coordinate Descent Summary

- Algorithms:
  1. Randomized Coordinate Descent
  2. Greedy (Steepest) Coordinate Descent
  3. Exact Coordinate Minimization
- Convergence Results:
  1. Randomized Coordinate Descent: Smooth Functions ($\frac{2dLR_0^2}{k}$

Coordinate Descent Summary

- Algorithms:
  1. Randomized Coordinate Descent
  2. Greedy (Steepest) Coordinate Descent
  3. Exact Coordinate Minimization
- Convergence Results:
  1. Randomized Coordinate Descent: Smooth Functions ($\frac{2dLR_0^2}{k}$
  2. Smooth + Strongly Convex: $(1 - \frac{\sigma}{dL})^k(f(x_0) - f_*)$

Coordinate Descent Summary

- Algorithms:
  1. Randomized Coordinate Descent
  2. Greedy (Steepest) Coordinate Descent
  3. Exact Coordinate Minimization
- Convergence Results:
  1. Randomized Coordinate Descent: Smooth Functions ($\frac{2dLR_0^2}{k}$
  2. Smooth + Strongly Convex: $(1 - \frac{\sigma}{dL})^k (f(x_0) - f_*)$
  3. Slight Improvement for Steepest CD: $(1 - \frac{\sigma}{dL})^k (f(x_0) - f_*)$

Coordinate Descent Summary

- Algorithms:
  1. Randomized Coordinate Descent
  2. Greedy (Steepest) Coordinate Descent
  3. Exact Coordinate Minimization
- Convergence Results:
  1. Randomized Coordinate Descent: Smooth Functions ($\frac{2dLR_0^2}{k}$
  2. Smooth + Strongly Convex: $(1 - \frac{\sigma}{dL})^k(f(x_0) - f_*)$
  3. Slight Improvement for Steepest CD: $(1 - \frac{\sigma}{dL})^k(f(x_0) - f_*)$
  4. Similar bound for exact coordinate minimization. Method of choice if the coordinate wise minimization can be solved exactly!

Coordinate Descent Summary

- Algorithms:
  1. Randomized Coordinate Descent
  2. Greedy (Steepest) Coordinate Descent
  3. Exact Coordinate Minimization
- Convergence Results:
  1. Randomized Coordinate Descent: Smooth Functions ($\frac{2dLR_0^2}{k}$
  2. Smooth + Strongly Convex: $(1 - \frac{\sigma}{dL})^k(f(x_0) - f_*)$
  3. Slight Improvement for Steepest CD: $(1 - \frac{\sigma}{dL})^k(f(x_0) - f_*)$
  4. Similar bound for exact coordinate minimization. Method of choice if the coordinate wise minimization can be solved exactly!
  5. Above bounds can also be extended if the non-differentiability is seperable (e.g. L1 regularization).

Coordinate Descent Summary

- Algorithms:
  1. Randomized Coordinate Descent
  2. Greedy (Steepest) Coordinate Descent
  3. Exact Coordinate Minimization
- Convergence Results:
  1. Randomized Coordinate Descent: Smooth Functions ($\frac{2dLR_0^2}{k}$
  2. Smooth + Strongly Convex: $(1 - \frac{\sigma}{dL})^k (f(x_0) - f_*)$
  3. Slight Improvement for Steepest CD: $(1 - \frac{\sigma}{dL})^k (f(x_0) - f_*)$
  4. Similar bound for exact coordinate minimization. Method of choice if the coordinate wise minimization can be solved exactly!
  5. Above bounds can also be extended if the non-differentiability is seperable (e.g. L1 regularization).
- Extensions: Accelerated Coordinate Minimization also possible!