Modern Optimization Techniques

Lucas Rego Drumond

Information Systems and Machine Learning Lab (ISMLL)
Institute of Computer Science
University of Hildesheim, Germany

Coordinate Descent

Outline

1. Review
2. Coordinate Descent
3. Examples

Outline

1. Review

2. Coordinate Descent

3. Examples

Gradient Descent

```
procedure GradientDescent
input: f_0
    Get initial point x
    repeat
        Get Step Size μ
        x := x − μ∇f_0(x)
    until convergence
    return x, f_0(x)
end procedure
```

Stochastic Gradient Descent (SGD)

```
procedure StochasticGradientDescent
input: f_0, μ
    Get initial point x
    repeat
        for i ∈ 1, ..., m do
            x → x − μ∇g(x, i)
        end for
    until convergence
    return x, f_0(x)
end procedure
```

Techniques to adjust the step size

- **Fixed step size**
- **Bold-Driver**
  - Evaluate the objective function $f_0(\mathbf{x}^t)$
  - if $f_0(\mathbf{x}^t) < f_0(\mathbf{x}^{t-1})$ then increase μ
  - else $f_0(\mathbf{x}^t) > f_0(\mathbf{x}^{t-1})$ then decrease μ
- **AdaGrad**
  - Update the gradient history $\mathbf{h} \rightarrow \mathbf{h} + \nabla_{\mathbf{x}} g(\mathbf{x}, i) \circ \nabla_{\mathbf{x}} g(\mathbf{x}, i)$
  - The step size for parameter x_i is $\frac{\mu}{\sqrt{h_i}}$

Outline

Coordinate Descent

- Gradient Descent and Stochastic Gradient Descent:
  - Use first order information (the gradient) to update *all* the variables
  - Because of that they belong to the class of First-Order Methods
- Instead of optimizing *all* the variables, we can optimize them *one at a time*

Coordinate wise minimization:
Suppose we want to minimize a function $f_0 : \mathbb{R}^n \to \mathbb{R}$

Find $\mathbf{x}_i^*$ for $i = 1, \ldots, n$ one at a time such that

$$f_0(\mathbf{x}^*) = \min f_0(\mathbf{x})$$

Can we do Coordinate-wise minimization?

Suppose we have a convex and differentiable function $f_0 : \mathbb{R}^n \to \mathbb{R}$

If we have a point $\mathbf{x}$ that is optimized (minimized) on each axis, is $\mathbf{x}$ a global optimum?

Question: Suppose $f_0(\mathbf{x}^*) = \min f_0(\mathbf{x})$, is it always true that:

$$f_0(\mathbf{x}^* + \mathbf{w} \circ \mathbf{e}^{(\mathbf{i})}) \geq f_0(\mathbf{x}^*)$$

for every $\mathbf{w} \in \mathbb{R}^n$ and $1 \leq i \leq n$?

Where:

- $\mathbf{e}^{(\mathbf{i})} \in \mathbb{R}^n$ is a standard basis vector such that: $e_i^{(i)} = 1$ and $e_j^{(i)} = 0$ for $i \neq j$
- $\circ$ denotes the elementwise product

Can we do Coordinate-wise minimization?

Question: Suppose $f_0(\mathbf{x}^*) = \min f_0(\mathbf{x})$, is it always true that:

$$f_0(\mathbf{x}^* + \mathbf{w} \circ \mathbf{e}^{(\mathbf{i})}) \geq f_0(\mathbf{x}^*)$$

for every $\mathbf{w} \in \mathbb{R}^n$ and $1 \leq i \leq n$?

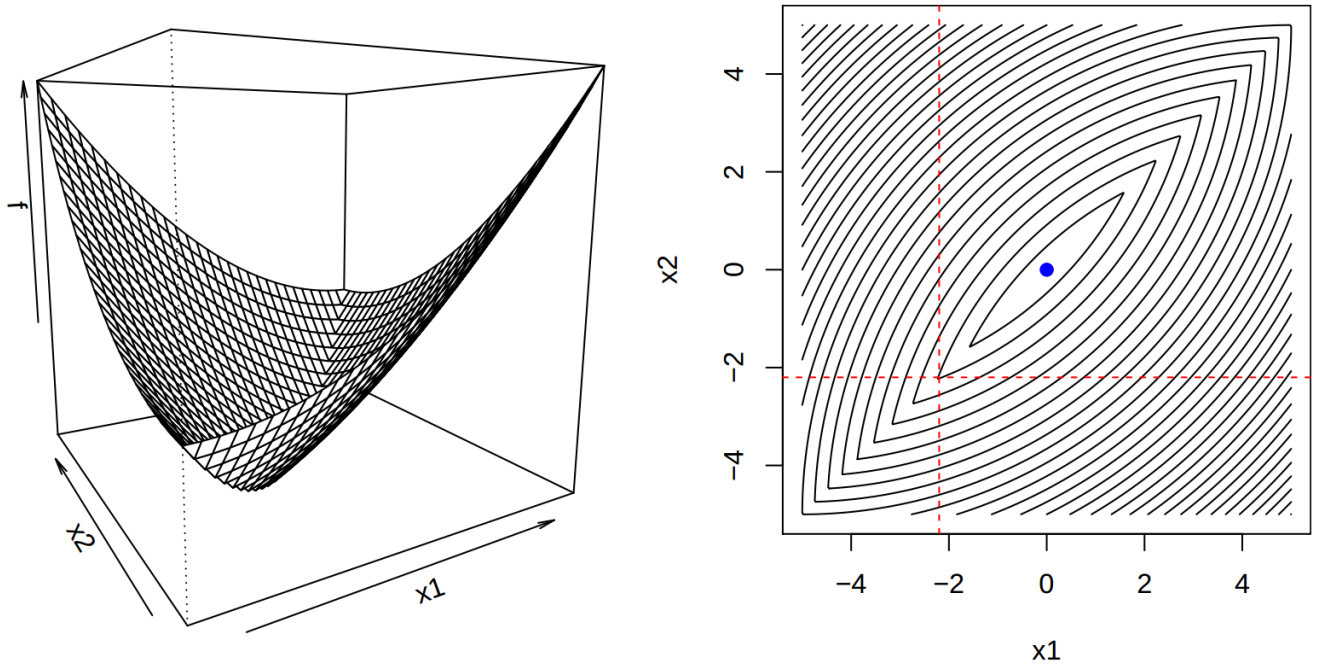
For f_0 **convex** and **differentiable**, if $\mathbf{x}^*$ minimizes f_0, then:

$$\nabla f_0(\mathbf{x}^*) = \left(\frac{\partial f_0(\mathbf{x}^*)}{\partial \mathbf{x}_1^*}, \frac{\partial f_0(\mathbf{x}^*)}{\partial \mathbf{x}_2^*}, \ldots, \frac{\partial f_0(\mathbf{x}^*)}{\partial \mathbf{x}_n^*} \right) = \mathbf{0}$$

but what about non-differentiable functions?

Can we do Coordinate-wise minimization?

Non-smooth function:



https://www.cs.cmu.edu/~ggordon/10725-F12/slides/25-coord-desc.pdf

Can we do Coordinate-wise minimization?

We actually can use Coordinate Descent on functions of the type:

$$f_0(\mathbf{x}) = g(\mathbf{x}) + \sum_{i=1}^{n} h_i(x_i)$$

where:

- g is smooth and convex
- each h_i is convex

Proof: $\mathbf{x}^*$ is the minimizer of f_0. for any α,

$$\begin{aligned} f_0(\alpha) - f_0(\mathbf{x}) &\geq \nabla g(\mathbf{x}^*)^T(\alpha - \mathbf{x}^*) + \sum_{i=1}^{n}[h_i(\alpha_i) - h_i(\mathbf{x}_i^*)] \\ &= \sum_{i=1}^{n}[\nabla_i g(\mathbf{x}^*)(\alpha_i - \mathbf{x}_i^*) + h_i(\alpha_i) - h_i(\mathbf{x}_i^*)] \geq 0 \end{aligned}$$

Coordinate Descent

We start with an initial guess $\mathbf{x}^{(0)}$

Repeat the following steps for $t = 1, 2, 3, \ldots$

$$\mathbf{x}_1^{(t)} \leftarrow \arg\min_{x_1} f_0(x_1, \mathbf{x}_2^{(t-1)}, \mathbf{x}_3^{(t-1)}, \ldots, \mathbf{x}_n^{(t-1)})$$

$$\mathbf{x}_2^{(t)} \leftarrow \arg\min_{x_2} f_0(\mathbf{x}_1^{(t)}, x_2, \mathbf{x}_3^{(t-1)}, \ldots, \mathbf{x}_n^{(t-1)})$$

$$\mathbf{x}_3^{(t)} \leftarrow \arg\min_{x_3} f_0(\mathbf{x}_1^{(t)}, \mathbf{x}_2^{(t)}, x_3, \ldots, \mathbf{x}_n^{(t-1)})$$

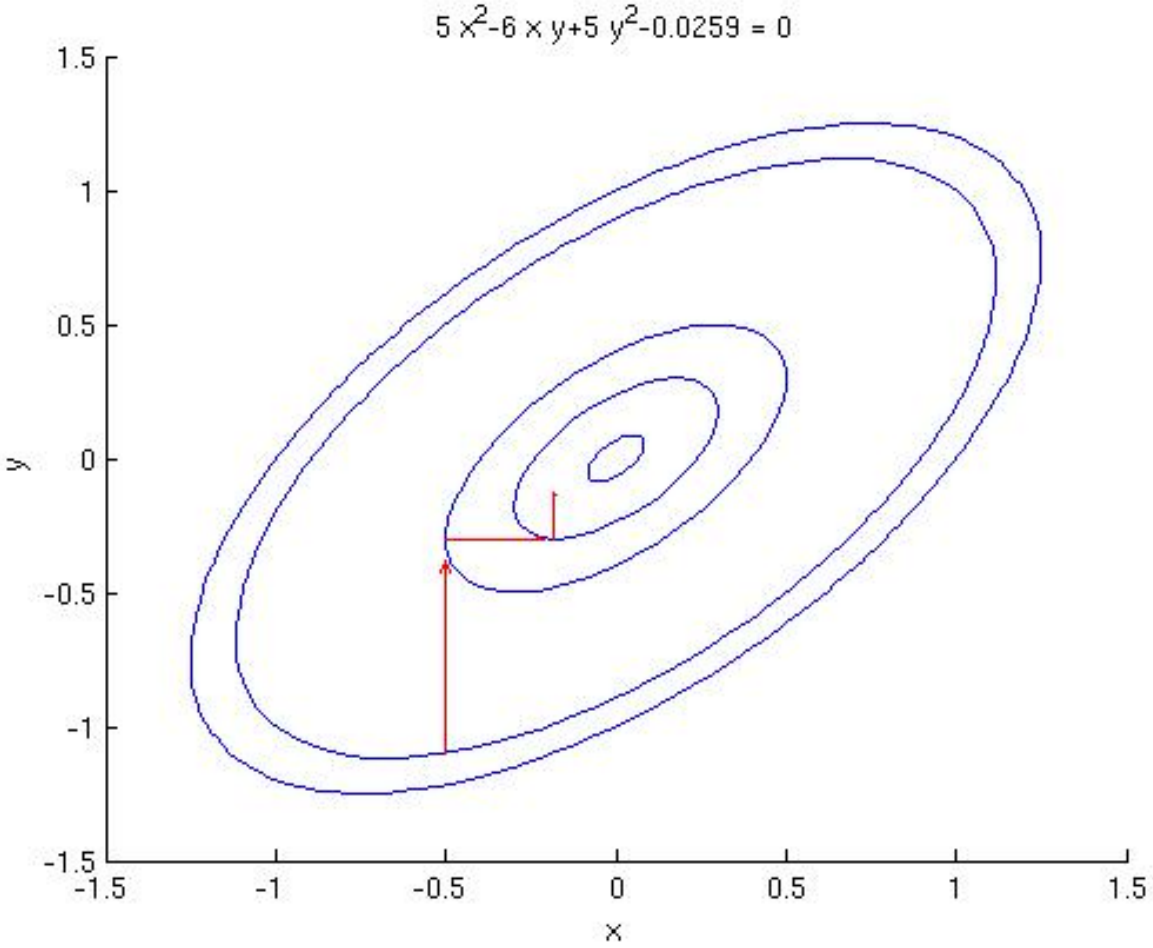
$$\vdots$$

$$\mathbf{x}_n^{(t)} \leftarrow \arg\min_{x_n} f_0(\mathbf{x}_1^{(t)}, \mathbf{x}_2^{(t)}, \mathbf{x}_3^{(t)}, \ldots, x_n)$$

Coordinate Descent Algorithm

procedure COORDINATEDESCENT
input: f_0
  Get initial point $\mathbf{x}^{(0)}$
  $t \leftarrow 1$
  repeat
    $\mathbf{x}_1^{(t)} \leftarrow \arg\min_{x_1} f_0(x_1, \mathbf{x}_2^{(t-1)}, \mathbf{x}_3^{(t-1)}, \ldots, \mathbf{x}_n^{(t-1)})$
    $\mathbf{x}_2^{(t)} \leftarrow \arg\min_{x_2} f_0(\mathbf{x}_1^{(t)}, x_2, \mathbf{x}_3^{(t-1)}, \ldots, \mathbf{x}_n^{(t-1)})$
    $\mathbf{x}_3^{(t)} \leftarrow \arg\min_{x_3} f_0(\mathbf{x}_1^{(t)}, \mathbf{x}_2^{(t)}, x_3, \ldots, \mathbf{x}_n^{(t-1)})$
    $\vdots$
    $\mathbf{x}_n^{(t)} \leftarrow \arg\min_{x_n} f_0(\mathbf{x}_1^{(t)}, \mathbf{x}_2^{(t)}, \mathbf{x}_3^{(t)}, \ldots, x_n)$
    $t \leftarrow t + 1$
  until convergence
  return $\mathbf{x}$, $f_0(\mathbf{x})$
end procedure

Coordinate Descent Algorithm



http://en.wikipedia.org/wiki/Coordinate_descent

Coordinate Descent - Considerations

- The order through in which we cycle through the coordinates is arbitrary
- We may update blocks of coordinates at a time instead of only one
- Updating all variables at a time (all-at-once) **does not** necessarily converge
- No need to adjust a step-size!!
- Works poorly with non-smooth functions

Outline

Coordinate Descent - Example

For $\mathbf{x} \in \mathbb{R}^2$

$$\min_{\mathbf{x}} \quad (a_1 x_1 - a_2 x_2 + a_3)^2$$

Algorithm:

- Initialize $\mathbf{x}_1^{(0)}, \mathbf{x}_2^{(0)}$
- Repeat until convergence:
  - $\mathbf{x}_1^{(t)} \leftarrow \arg\min_{x_1} (a_1 x_1 - a_2 \mathbf{x}_2^{(t-1)} + a_3)^2$
  - $\mathbf{x}_2^{(t)} \leftarrow \arg\min_{x_2} (a_1 \mathbf{x}_1^{(t)} - a_2 x_2 + a_3)^2$

Coordinate Descent - Example

$$\arg\min_{x_1}(a_1 x_1 - a_2 x_2 + a_3)^2$$

Find a closed form solution:

$$\frac{d}{dx_1}(a_1 x_1 - a_2 x_2 + a_3)^2 \stackrel{!}{=} 0$$

$$\frac{d}{dx_1}(a_1 x_1 - a_2 x_2 + a_3)^2 = 2(a_1 x_1 - a_2 x_2 + a_3)a_1$$

$$2(a_1 x_1 - a_2 x_2 + a_3)a_1 \stackrel{!}{=} 0$$

$$2a_1^2 x_1 - 2a_1 a_2 x_2 + 2a_1 a_3 = 0$$

$$2a_1^2 x_1 = 2a_1 a_2 x_2 - 2a_1 a_3$$

$$x_1 = \frac{a_2 x_2 - a_3}{a_1}$$

Coordinate Descent - Example

$$\arg\min_{x_2}(a_1x_1 - a_2x_2 + a_3)^2$$

Find a closed form solution:

$$\frac{d}{dx_2}(a_1x_1 - a_2x_2 + a_3)^2 \stackrel{!}{=} 0$$

$$\frac{d}{dx_2}(a_1x_1 - a_2x_2 + a_3)^2 = -2(a_1x_1 - a_2x_2 + a_3)a_2$$

$$-2(a_1x_1 - a_2x_2 + a_3)a_2 \stackrel{!}{=} 0$$

$$-2a_1a_2x_1 + 2a_2^2x_2 - 2a_2a_3 = 0$$

$$2a_2^2x_2 = 2a_1a_2x_1 + 2a_2a_3$$

$$x_2 = \frac{a_1x_1 + a_3}{a_2}$$

Coordinate Descent - Example

For $\mathbf{x} \in \mathbb{R}^2$

$$\min_{\mathbf{x}} \quad (a_1 x_1 - a_2 x_2 + a_3)^2$$

Algorithm:

- Initialize $\mathbf{x}_1^{(0)}, \mathbf{x}_2^{(0)}$
- Repeat until convergence:
  - $\mathbf{x}_1^{(t)} \leftarrow \frac{a_2 \mathbf{x}_2^{(t-1)} - a_3}{a_1}$
  - $\mathbf{x}_2^{(t)} \leftarrow \frac{a_1 \mathbf{x}_1^{(t)} + a_3}{a_2}$

Coordinate Descent - Example

For $\mathbf{x} \in \mathbb{R}^2$, $a_1 = 0.1$, $a_2 = 2$, $a_3 = 1$

$$\min_{\mathbf{x}} \quad (0.1x_1 - 2x_2 + 1)^2$$

$$\mathbf{x}_1^{(t)} \leftarrow \frac{2\mathbf{x}_2^{(t-1)} - 1}{0.1} \qquad \mathbf{x}_2^{(t)} \leftarrow \frac{0.1\mathbf{x}_1^{(t)} + 1}{2}$$

Start with $\mathbf{x}_1^{(0)} = 1, \mathbf{x}_2^{(0)} = 2$

- $\mathbf{x}_1^{(1)} \leftarrow \frac{2 \cdot 2 - 1}{0.1} = 30$
- $\mathbf{x}_2^{(1)} \leftarrow \frac{0.1 \cdot 30 + 1}{2} = 2$

$\mathbf{x}_1^{(1)} = 30, \mathbf{x}_2^{(1)} = 2$

- $\mathbf{x}_1^{(2)} \leftarrow \frac{2 \cdot 2 - 1}{0.1} = 30$
- $\mathbf{x}_2^{(2)} \leftarrow \frac{0.1 \cdot 30 + 1}{2} = 2$

Coordinate Descent for Linear Regression

For the problem

$$\text{minimize} \quad \sum_{i=1}^{m}(y_i - \mathbf{x}^T\mathbf{a_i})^2 = \sum_{i=1}^{m}\left(y_i - \sum_{j=1}^{n} x_j a_{ij}\right)^2$$

We can compute the update rule for a specific x_k:

$$\frac{\partial f_0(\mathbf{x})}{\partial x_k} \stackrel{!}{=} 0$$

$$\frac{\partial f_0(\mathbf{x})}{\partial x_k} = 2 \cdot \sum_{i=1}^{m} a_{ik} \cdot \left(y_i - \sum_{j=1}^{n} x_j a_{ij}\right)$$

Coordinate Descent for Linear Regression

$$2 \cdot \sum_{i=1}^{m} a_{ik} \cdot \left(y_i - \sum_{j=1}^{n} x_j a_{ij} \right) = 0$$

$$\sum_{i=1}^{m} a_{ik} \cdot y_i - \sum_{i=1}^{m} a_{ik} \cdot \sum_{j=1}^{n} x_j a_{ij} = 0$$

$$\sum_{i=1}^{m} a_{ik} \cdot y_i - \sum_{i=1}^{m} a_{ik} \cdot \left(x_k a_{ik} + \sum_{j=1,\, j \neq k}^{n} x_j a_{ij} \right) = 0$$

$$\sum_{i=1}^{m} a_{ik} \cdot y_i - \sum_{i=1}^{m} a_{ik} \cdot x_k a_{ik} - \sum_{i=1}^{m} a_{ik} \cdot \sum_{j=1,\, j \neq k}^{n} x_j a_{ij} = 0$$

$$\sum_{i=1}^{m} a_{ik} \cdot y_i - x_k \sum_{i=1}^{m} a_{ik}^2 - \sum_{i=1}^{m} a_{ik} \cdot \sum_{j=1,\, j \neq k}^{n} x_j a_{ij} = 0$$

Coordinate Descent for Linear Regression

$$\sum_{i=1}^{m} a_{ik} \cdot y_i - x_k \sum_{i=1}^{m} a_{ik}^2 - \sum_{i=1}^{m} a_{ik} \cdot \sum_{j=1,\, j \neq k}^{n} x_j a_{ij} = 0$$

$$x_k \cdot \sum_{i=1}^{m} a_{ik}^2 = \sum_{i=1}^{m} a_{ik} \cdot y_i - \sum_{i=1}^{m} a_{ik} \cdot \sum_{j=1,\, j \neq k}^{n} x_j a_{ij}$$

$$x_k = \frac{\sum_{i=1}^{m} a_{ik} \cdot \left(y_i - \sum_{j=1,\, j \neq k}^{n} x_j a_{ij} \right)}{\sum_{i=1}^{m} a_{ik}^2}$$

Linear Regression Coordinate Descent - Simple Algorithm

procedure LINEAR REGRESSION-CD
input: f_0
    Get initial point $\mathbf{x}^{(0)}$
    repeat
        for $k \in 1, \ldots n$ **do**
            $x_k \leftarrow \frac{\sum_{i=1}^{m} a_{ik} \cdot \left(y_i - \sum_{j=1,\, j \neq k}^{n} x_j a_{ij}\right)}{\sum_{i=1}^{m} a_{ik}^2}$
        end for
    until convergence
    return $\mathbf{x}$, $f_0(\mathbf{x})$
end procedure

Coordinate Descent for Linear Regression

For each parameter we have the following update rule:

$$x_k = \frac{\sum_{i=1}^{m} a_{ik} \cdot \left(y_i - \sum_{j=1,\, j \neq k}^{n} x_j a_{ij} \right)}{\sum_{i=1}^{m} a_{ik}^2}$$

One Coordinate descent epoch has a cost of $O(m \cdot n^2)$!

Can we do it faster?

CD for Linear Regression - Smart Update

For each parameter we have the following update rule:

$$x_k = \frac{\sum_{i=1}^{m} a_{ik} \cdot \left(y_i - \sum_{j=1,\, j \neq k}^{n} x_j a_{ij} \right)}{\sum_{i=1}^{m} a_{ik}^2}$$

We can rewrite:

$$\begin{aligned}
\sum_{i=1}^{m} \left(y_i - \sum_{j=1,\, j \neq k}^{n} x_j a_{ij} \right) &= \sum_{i=1}^{m} \left(y_i - \sum_{j=1}^{n} x_j a_{ij} + x_k a_{ik} \right) \\
&= \sum_{i=1}^{m} \left(y_i - \sum_{j=1}^{n} x_j a_{ij} \right) + \sum_{i=1}^{m} x_k a_{ik} \\
&= \sum_{i=1}^{m} \left(y_i - \sum_{j=1}^{n} x_j a_{ij} \right) + x_k \sum_{i=1}^{m} a_{ik}
\end{aligned}$$

CD for Linear Regression - Smart Update

From which we have:

$$
\begin{aligned}
x_k &= \frac{\sum_{i=1}^{m} a_{ik} \cdot \left(y_i - \sum_{j=1,\, j \neq k}^{n} x_j a_{ij} \right)}{\sum_{i=1}^{m} a_{ik}^2} \\
&= \frac{\sum_{i=1}^{m} a_{ik} \cdot \left(y_i - \sum_{j=1}^{n} x_j a_{ij} + \mathbf{x}_k^{old} a_{ik} \right)}{\sum_{i=1}^{m} a_{ik}^2} \\
&= \frac{\sum_{i=1}^{m} a_{ik} \cdot \left(y_i - \sum_{j=1}^{n} x_j a_{ij} \right)}{\sum_{i=1}^{m} a_{ik}^2} + \frac{\sum_{i=1}^{m} a_{ik} \cdot \left(\mathbf{x}_k^{old} a_{ik} \right)}{\sum_{i=1}^{m} a_{ik}^2} \\
&= \frac{\sum_{i=1}^{m} a_{ik} \cdot \left(y_i - \sum_{j=1}^{n} x_j a_{ij} \right)}{\sum_{i=1}^{m} a_{ik}^2} + \frac{\mathbf{x}_k^{old} \cdot \sum_{i=1}^{m} a_{ik}^2}{\sum_{i=1}^{m} a_{ik}^2} \\
&= \frac{\sum_{i=1}^{m} a_{ik} \cdot \left(y_i - \sum_{j=1}^{n} x_j a_{ij} \right)}{\sum_{i=1}^{m} a_{ik}^2} + \mathbf{x}_k^{old}
\end{aligned}
$$

CD for Linear Regression - Smart Update

Now we have:

$$x_k = \frac{\sum_{i=1}^{m} a_{ik} \cdot \left(y_i - \sum_{j=1}^{n} x_j a_{ij} \right)}{\sum_{i=1}^{m} a_{ik}^2} + \mathbf{x}_k^{old}$$

So we can define our residual vector $\mathbf{r} \in \mathbb{R}^m$ such that

$$r_i = y_i - \sum_{j=1}^{n} x_j a_{ij}$$

CD for Linear Regression - Smart Update

After each update of x_k, we can maintain r_i instead of recomputing it given the old value $\mathbf{x}_k^{old}$:

$$
\begin{aligned}
r_i^{new} &= y_i - \left(\sum_{j=1}^{n} x_j a_{ij} - \mathbf{x}_k^{old} a_{ik} + x_k a_{ik} \right) \\
&= r_i^{old} + (\mathbf{x}_k^{old} - x_k) a_{ik}
\end{aligned}
$$

CD for Linear Regression - Smart Update

Now our algorithm looks like:

1. Initialize $\mathbf{x}$
2. Compute $r_i = y_i - \sum_{j=1}^{n} x_j a_{ij}$
3. While Not Converged
   3.1 For each $k = 1, \ldots, n$
      3.1.1 $\mathbf{x}_k^{old} \leftarrow x_k$
      3.1.2 $x_k \leftarrow \frac{\sum_{i=1}^{m} a_{ik} \cdot (r_i)}{\sum_{i=1}^{m} a_{ik}^2} + \mathbf{x}_k^{old}$
      3.1.3 For all i $r_i \leftarrow r_i + (\mathbf{x}_k^{old} - x_k) a_{ik}$

This algorithm is now $O(m \cdot n)$!

Linear Regression Coordinate Descent Algorithm

procedure LINEAR REGRESSION-CD
input: f_0
  Get initial point $\mathbf{x}^{(0)}$
  $\mathbf{r} \leftarrow \mathbf{y} - A\mathbf{x}^{(0)}$
  repeat
    for $k \in 1, \dots n$ **do**
      $\mathbf{x}_k^{old} \leftarrow x_k$
      $x_k \leftarrow \frac{\sum_{i=1}^{m} a_{ik} \cdot r_i}{\sum_{i=1}^{m} a_{ik}^2} + \mathbf{x}_k^{old}$
      for $i \in 1, \dots m$ **do**
        $r_i \leftarrow r_i + (\mathbf{x}_k^{old} - x_k) a_{ik}$
      end for
    end for
  until convergence
  return $\mathbf{x}$, $f_0(\mathbf{x})$
end procedure

Real World Dataset: Body Fat prediction

We want to estimate the percentage of body fat based on various attributes:

- Age (years)
- Weight (lbs)
- Height (inches)
- Neck circumference (cm)
- Chest circumference (cm)
- Abdomen 2 circumference (cm)
- Hip circumference (cm)
- Thigh circumference (cm)
- Knee circumference (cm)
- ...

`http://lib.stat.cmu.edu/datasets/bodyfat`

Real World Dataset: Body Fat prediction

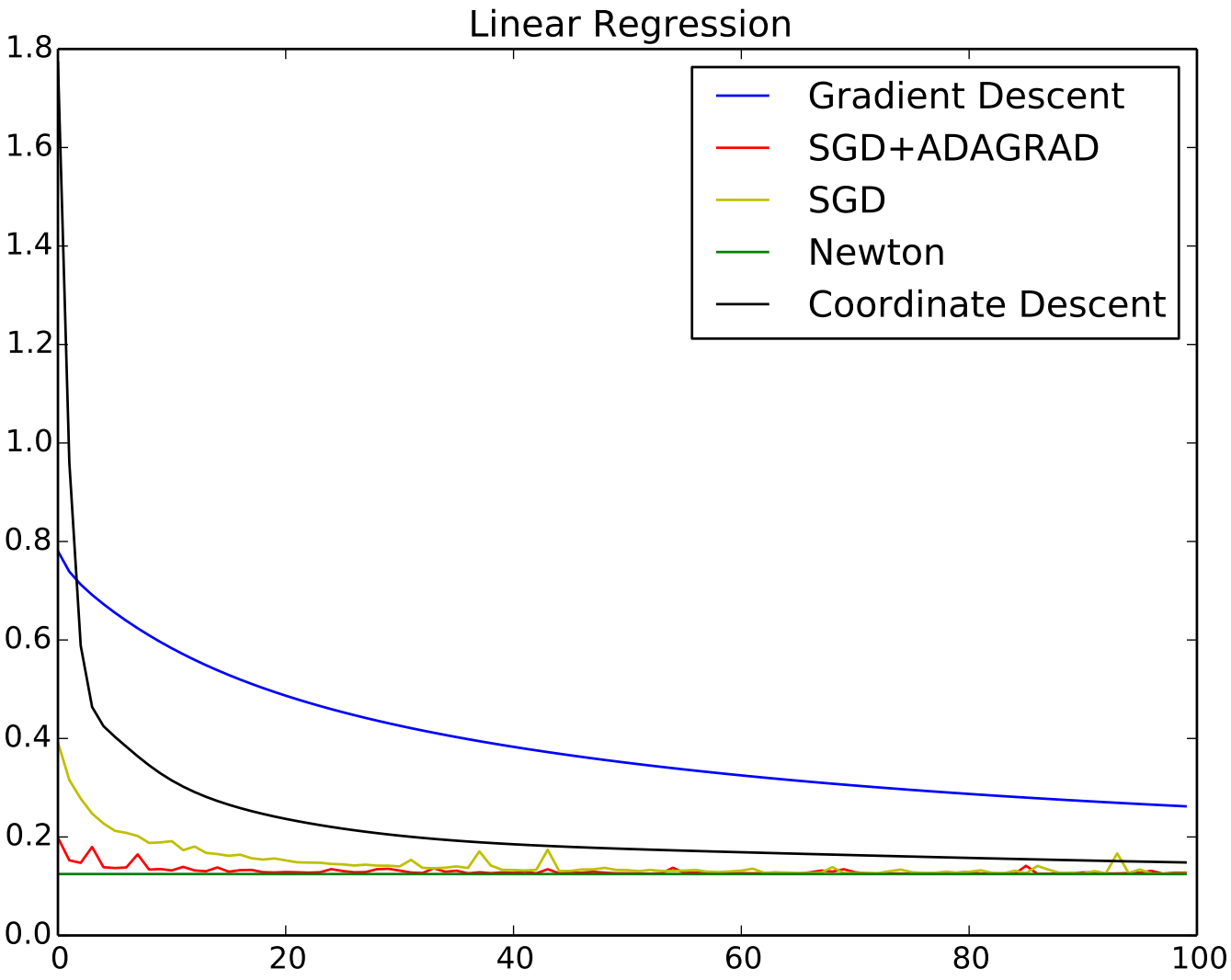
The data is represented it as:

$$A_{m,n} = \begin{pmatrix} 1 & a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ 1 & a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & a_{m,1} & a_{m,2} & \dots & a_{m,n} \end{pmatrix} \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}$$

with $m = 252$, $n = 14$
We can model the percentage of body fat y is a linear combination of the body measurements with parameters $\mathbf{x}$:

$$\hat{y}_i = \mathbf{x}^T \mathbf{a_i} = x_0 1 + x_1 a_{i,1} + x_2 a_{i,2} + \ldots + x_n a_{i,n}$$

Coordinate Descent - Body fat dataset



Year Prediction Data Set

- Least Squares Problem
- Prediction of the release year of a song from audio features
- 90 features
- Experiments done on a subset of 1000 instances of the data

Coordinate Descent - Year Prediction