Math 7243

Machine Learning and Statistical Learning Theory

Section 4. Gradient Descent

Instructor: He Wang
Department of Mathematics
Northeastern University

Review :

Today:

1. Gradient Decent
2. Stochastic Gradient Decent
3. Newton's Method

Taylor Expension of $f: \mathbb{R} \to \mathbb{R}$

$$f(x+s) = f(x) + s f'(x) + \frac{1}{2} s^2 f''(x) + \cdots$$

Taylor Expansion of $F: \mathbb{R}^d \to \mathbb{R}$

$$F(\vec{x}+\vec{s}) = F(\vec{s}) + \vec{s}^T \nabla F + \frac{1}{2} \vec{s}^T H s + \cdots$$

where H is the Hessian Matrix

➢ Gradient Descent

Goal: find the local/global minimum of the cost function. $J(\vec{\theta})$

1. No closed formula or too complicated to find a closed formula for the minimum.
2. Too complicated to compute even we have a formula, as the inverse.

Suppose $f(\vec{x})$ is a differentiable function. Which direction has the largest rate of change?

Theorem. (Calculus)

The maximum value of the directional derivative $D_{\vec{u}} f(\vec{x})$ is $|\nabla f(\vec{x})|$ and it occurs when $\vec{u}$ has the same direction as the gradient vector $\nabla f(\vec{x})$.

rate of change of $f(\vec{x})$ in direction $\vec{u}$

$|\vec{u}|=1$

$\nabla f = \begin{bmatrix} f_{x_1} \\ \vdots \\ f_{x_d} \end{bmatrix}$

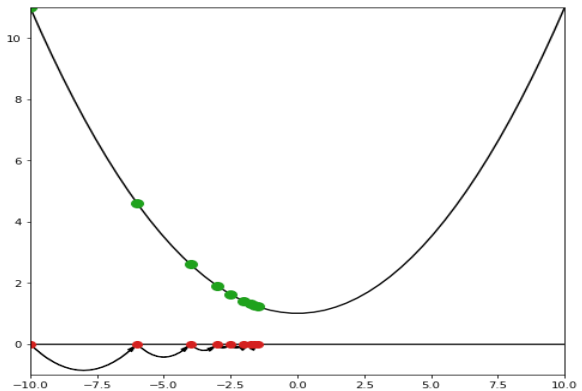
$$D_{\vec{u}}(f(\vec{x})) = (\nabla f)\cdot \vec{u} = |\nabla f|\cdot|\vec{u}|\cos\theta$$

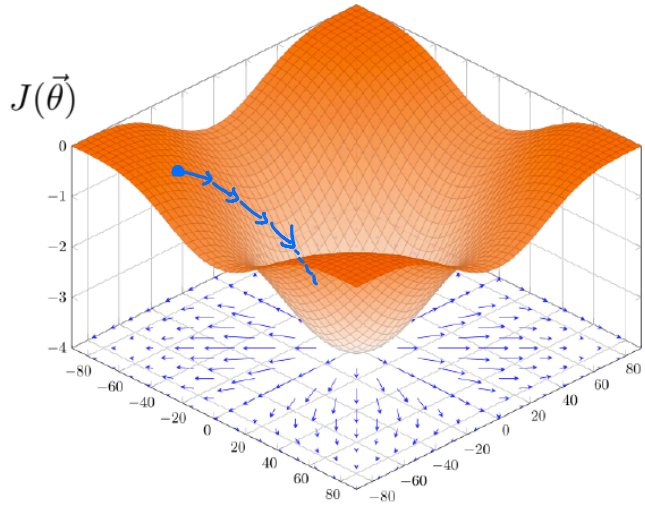
$$= |\nabla f|\cos\theta$$

$$\leq |\nabla f|$$

equality holds when $\theta = 0$

- minimum value in direction $-\nabla f$





- Gradient Descent: for $J(\vec{\theta})$

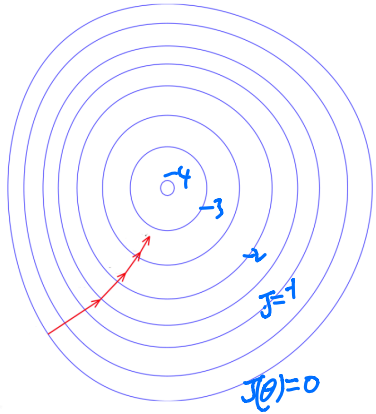
- Start with $\vec{\theta} =$ some initial value.

- Repeat $\vec{\theta} := \vec{\theta} - \alpha \nabla J(\vec{\theta})$ until converge.

next

learning rate

$$\begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} := \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} - \alpha \begin{bmatrix} \frac{\partial J(\vec{\theta})}{\partial \theta_0} \\ \frac{\partial J(\vec{\theta})}{\partial \theta_1} \\ \vdots \\ \frac{\partial J(\vec{\theta})}{\partial \theta_d} \end{bmatrix}$$

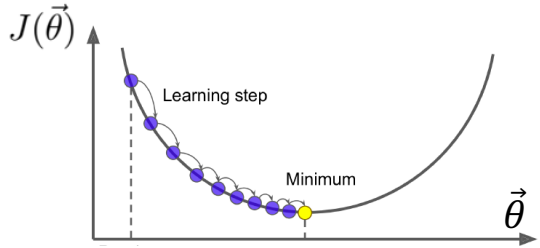


For each $j = 0, 1, \cdots, d$,

repeat $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} (J(\vec{\theta}))$ until converge.

keys: ① compute ∇J

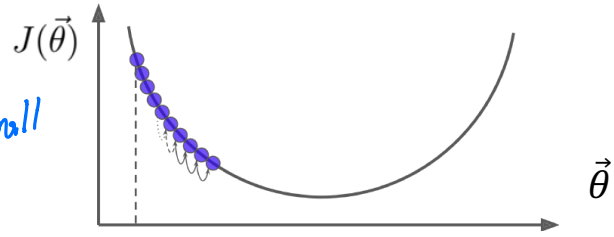
② Set initial value $\vec{\theta} = \vec{0} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$



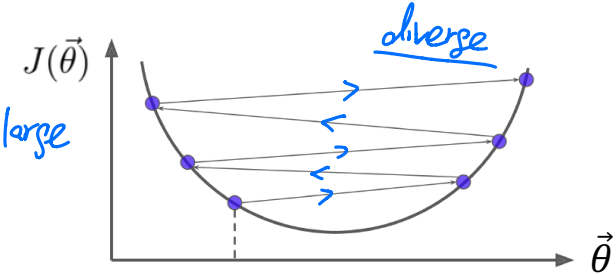
③ Set a "good" α.

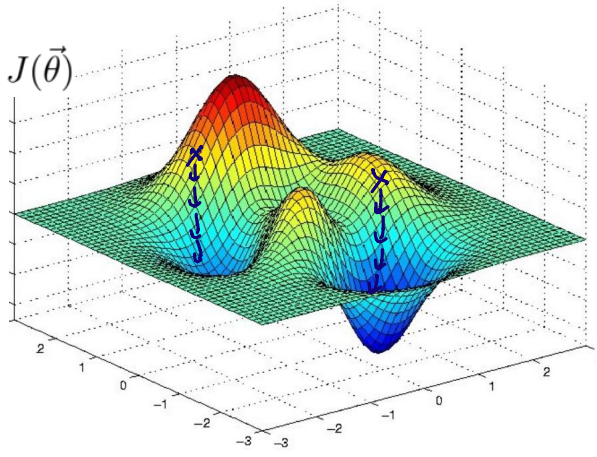
- If α is too small, θ_j converges slowly.
- If α is too large, θ_j may even diverge.
- Different initial value may result in different local minimum.

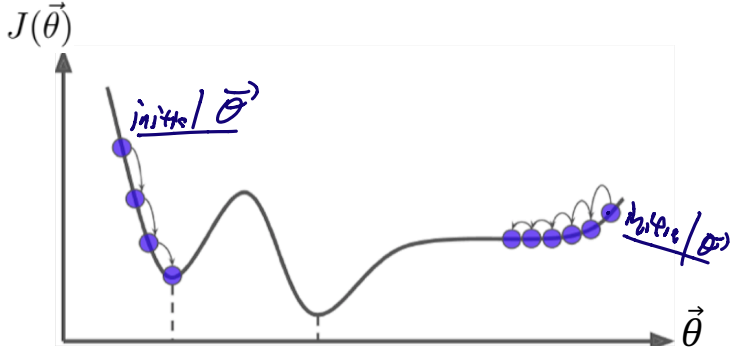
α too small



α to large







We ran the update rule for all the training examples at once, which is called **(batch) gradient descent.**

$(x^{(i)}, y^{(i)})$ $i = 1, 2, \cdots, n$.

Notation: $x^{(i)}$ row vector

- Example: (linear regression) $h(\vec{x}) = \vec{\theta}^T \vec{x} = \theta_0 + \theta_1 x_1 + \cdots + \theta_d x_d$

$$J(\vec{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \left(h(x^{(i)}) - y^{(i)} \right)^2$$

for each $j = 0, 1, \ldots, d$

$$\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\vec{\theta}) &= \frac{\partial}{\partial \theta_j} \left(\frac{1}{n} \sum_{i=1}^{n} \left(h(x^{(i)}) - y^{(i)} \right)^2 \right) \\
&= \frac{1}{n} \left(\sum_{i=1}^{n} \frac{\partial}{\partial \theta_j} \left(h(x^{(i)}) - y^{(i)} \right)^2 \right) \\
&= \frac{1}{n} \cdot \sum_{i=1}^{n} \left(2(h(x^{(i)}) - y^{(i)}) \cdot \frac{\partial}{\partial \theta_j} (h(x^{(i)}) - y^{(i)}) \right) \\
&= \frac{2}{n} \sum_{i=1}^{n} \left((h(x^{(i)}) - y^{(i)}) \cdot \frac{\partial}{\partial \theta_j} \left(\theta_0 + \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)} + \ldots + \theta_d x_d^{(i)} - y^{(i)} \right) \right) \\
&= \frac{2}{n} \sum_{i=1}^{n} (h(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}
\end{aligned}$$

For each $j = 0, 1, \cdots, d$

Repeat until converge.

$$\theta_j^{\text{next}} := \theta_j - \alpha \cdot \left(\frac{2}{n} \sum_{i=1}^{n} (h(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)} \right)$$

or vector notation: $\vec{\theta} = \vec{\theta} - \alpha \left(\frac{2}{n}\right) \sum_{i=1}^{n} \left(h(x^{(i)}) - y^{(i)}\right) \vec{x}^{(i)T}$

$\in \mathbb{R}$

$X^T(X\vec{\theta} - \vec{y})$

$\vec{y} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$ $\begin{bmatrix} h(x^{(1)}) \\ h(x^{(2)}) \\ \vdots \\ h(x^{(n)}) \end{bmatrix} = X\vec{\theta}$

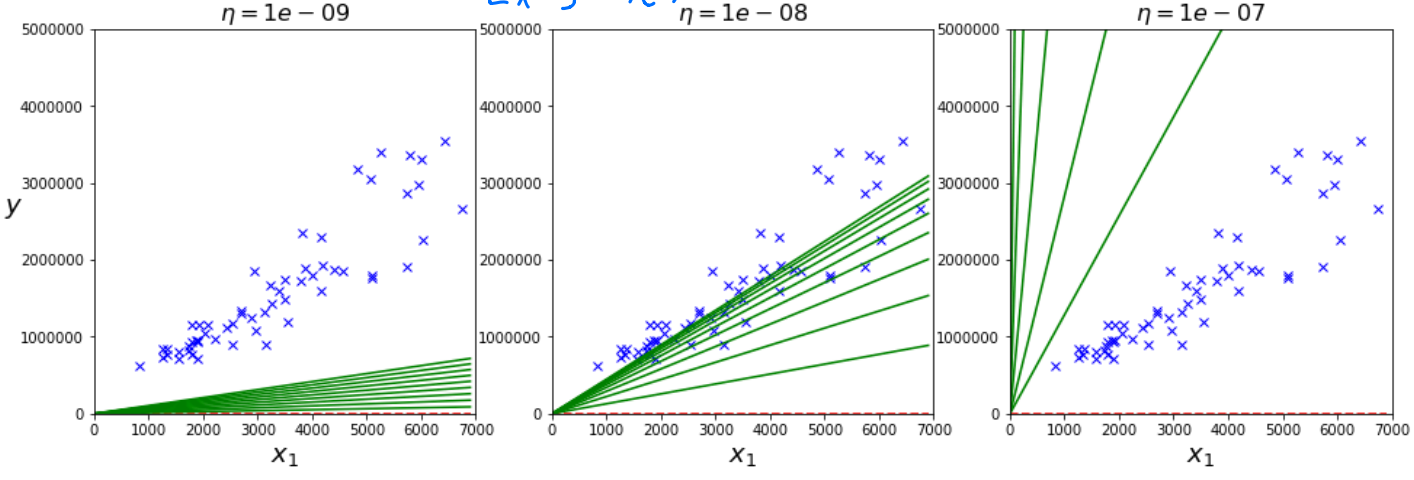
$X = \begin{bmatrix} x^{(1)} \\ \vdots \\ x^{(n)} \end{bmatrix}$ $n \times (d+1)$

Code:

$$\vec{\theta} = \vec{\theta} - \alpha \left(\frac{2}{n}\right) \text{sum}\left[\left(X\vec{\theta} - \vec{y}\right) * X \right]$$

$n \times 1$ $n \times (d+1)$

sum of rows element wise



- Example (Logistic regression) $h_{\vec{\theta}}(x) = S(\vec{\theta}^T \vec{x}) = \dfrac{1}{1 + e^{-\vec{\theta}^T \vec{x}}}$

$$J(\vec{\theta}) = \sum_{i=1}^{n} \left(h(x^{(i)}) - y^{(i)} \right)^2$$

- Stochastic Gradient Descent:

Repeat

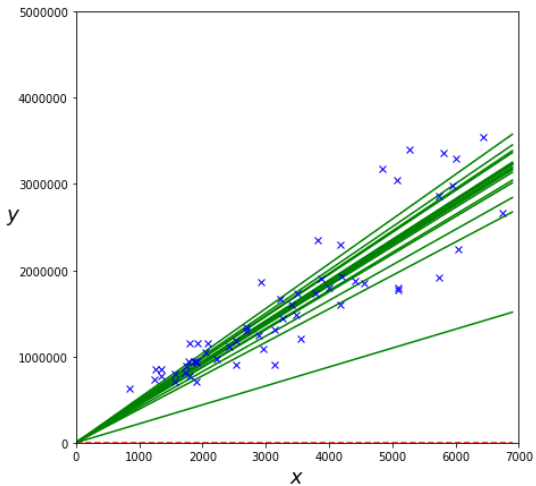
For i=1 to n {

For j=0 to d {

$$\theta_j := \theta_j - \alpha(\frac{2}{n}) \left(h(x^{(i)}) - y^{(i)}\right) x_j^{(i)}$$

}

}



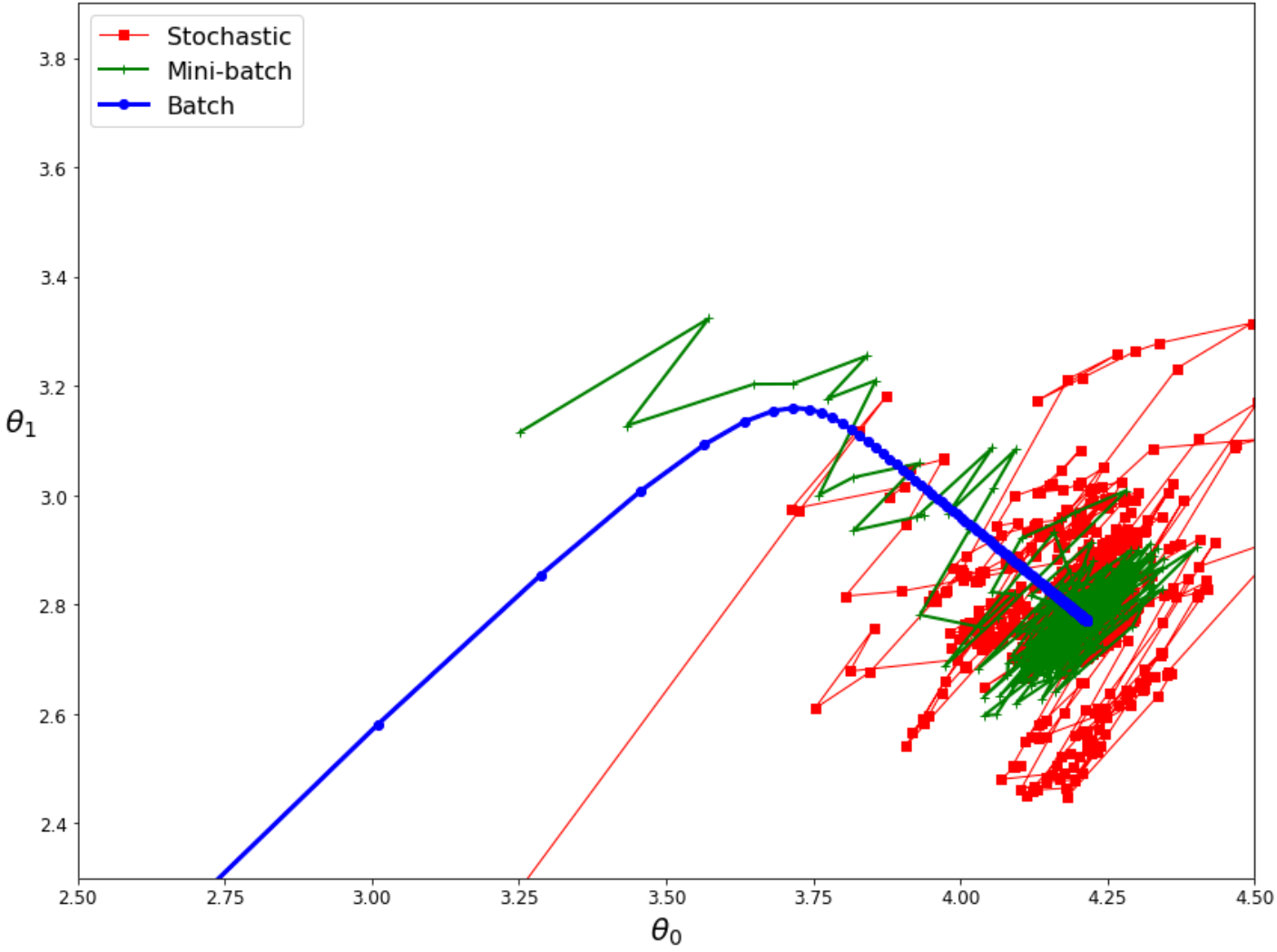
- Mini-batch Gradient Descent: batch (all training set) each step. + stochastic (one sample)

  • on a small random subset. each step.

Summary:

1. Normal equation — large n, small d, no scaling,
2. batch GD — small n, large d.
3. Stochastic GD — large n, large d
4. minibatch. — —

scikit-learn


Stochastic
Mini-batch
Batch
3.8
3.6
3.4
3.2
3.0
2.8
2.6
2.4
θ_1
2.50
2.75
3.00
3.25
3.50
3.75
4.00
4.25
4.50
θ_0

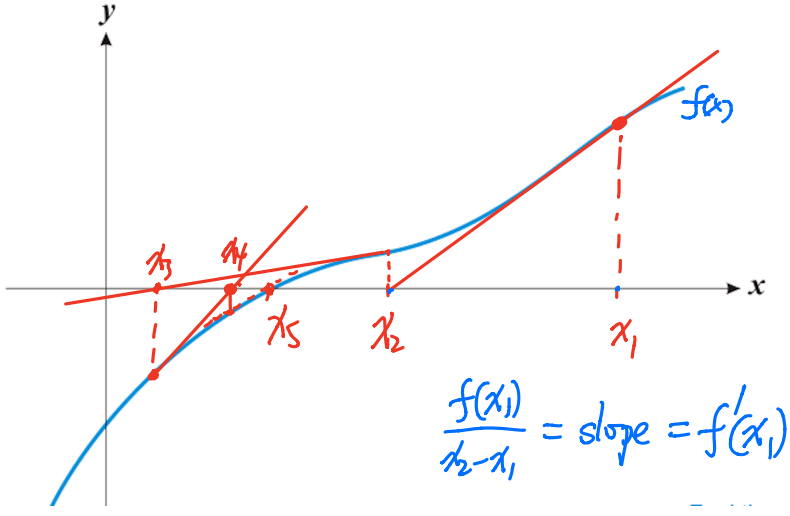
- Newton' method

Fund root of a function f(x).

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

$$\vdots$$

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$



$$\frac{f(x_1)}{x_2 - x_1} = \text{slope} = f'(x_1)$$

For $F(\vec{x}): R^n \longrightarrow R^n$

$$\vec{x}_{k+1} = \vec{x}_k - B^{-1} \cdot F(\vec{x})$$

Jacobian matrix

where $B = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$

Reason: $f(x_1 + \Delta) \approx f(x_1) + \Delta f'(x_1) = 0$

$$\Delta = -\frac{f(x_1)}{f'(x_1)}$$

Maximize $J(\theta)$

Want gradient $\nabla J(\theta) = \begin{bmatrix} \frac{\partial J}{\partial \theta_0} \\ \vdots \\ \frac{\partial J}{\partial \theta_d} \end{bmatrix} : \mathbb{R}^{d+1} \to \mathbb{R}^{d+1}$

$= 0$

$$\vec{\theta}_{(k+1)} = \vec{\theta}_{(k)} - H^{-1} \nabla_\theta J(\theta)$$

Single variable θ.

$$\theta_{(k+1)} = \theta_{(k)} - \frac{J'(\theta)}{J''(\theta)}$$

H is the Hessian matrix $H(J(\theta)) = \begin{bmatrix} \frac{\partial^2 J(\theta)}{\partial \theta_0^2} & \frac{\partial^2 J(\theta)}{\partial \theta_0 \partial \theta_1} & \cdots & \frac{\partial J(\theta)}{\partial \theta_0 \partial \theta_d} \\ & \cdots & & \\ \frac{\partial^2 J(\theta)}{\partial \theta_d \partial \theta_0} & & \cdots & \frac{\partial^2 J(\theta)}{\partial \theta_d \partial \theta_d} \end{bmatrix}$