Convex Optimization - Gradient Descent

Optimization

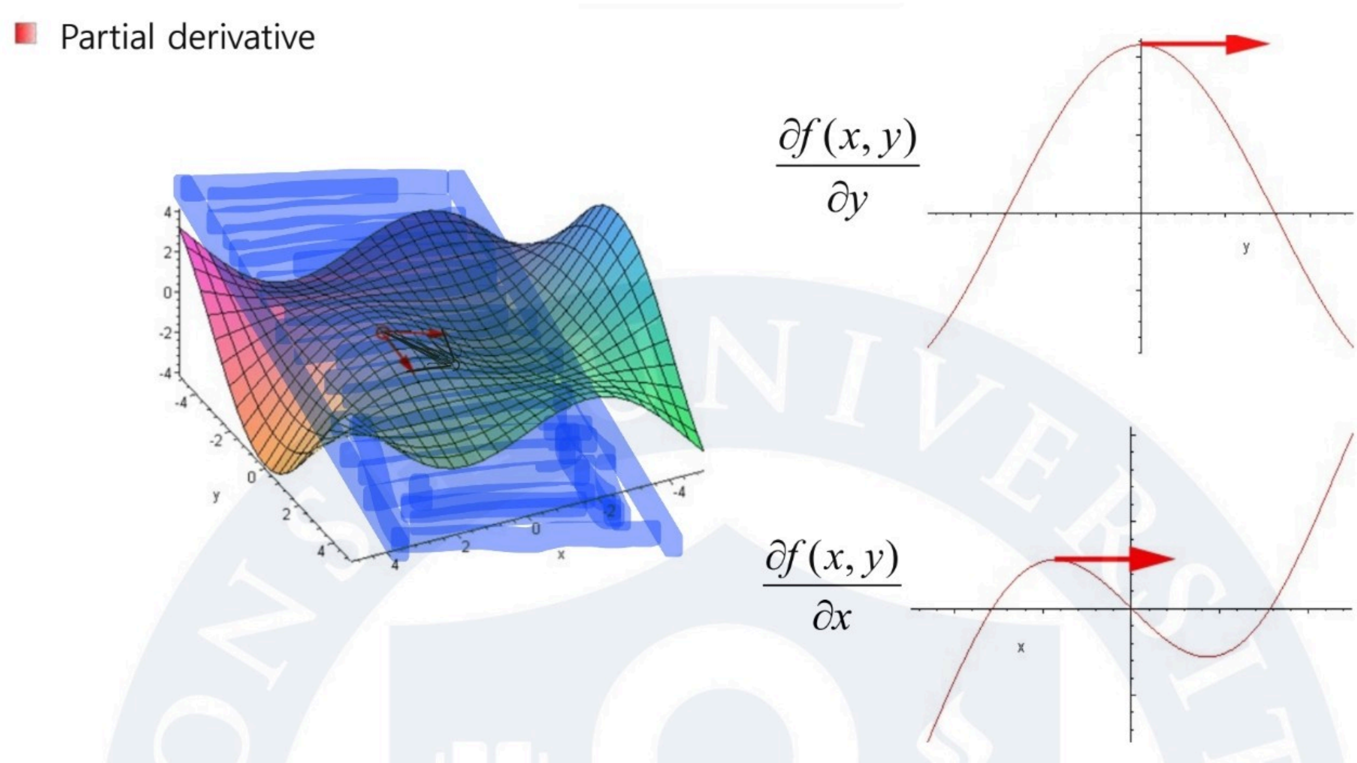
- Find value of decision variables that make object function have minimum(maximum) value.
- In machine learning, object function becomes Cost function which have to be minimized.
- e.g. : MSE, Entrophy, etc.

Gradient

- $\nabla f(x)$: Vector whose components are partial derivatives of f at some point x

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1}, \cdots, \frac{\partial f}{\partial x_p}\right)^T$$

- Which can be interpreted as **direction and magnitude in which the function moves**
- If we want to minimize objective function, We have to update variable in **negative direction of derivatives!!**



Algorithm

- Objective : $\min_{\theta} J(\theta)$
- Parameter : $\theta = [\theta_0. \theta_1 \cdots \theta_p]$
- Algorithm
  1. Set $t = 0$ and choose the learning rate(step size) α
  2. Compute $\nabla J(\theta)$
  3. Set $\theta^{(t+1)} := \theta^{(t)} - \alpha \nabla J(\theta)$
  4. Repeat 2~3 until $\nabla J(\theta) = 0$

Example - linear regression

- Functions

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2$$
$$\hat{y}_i = \theta_0 + \theta_1 x_{i1} + \cdots + \theta_p x_{ip}$$

- where $j \neq 0$

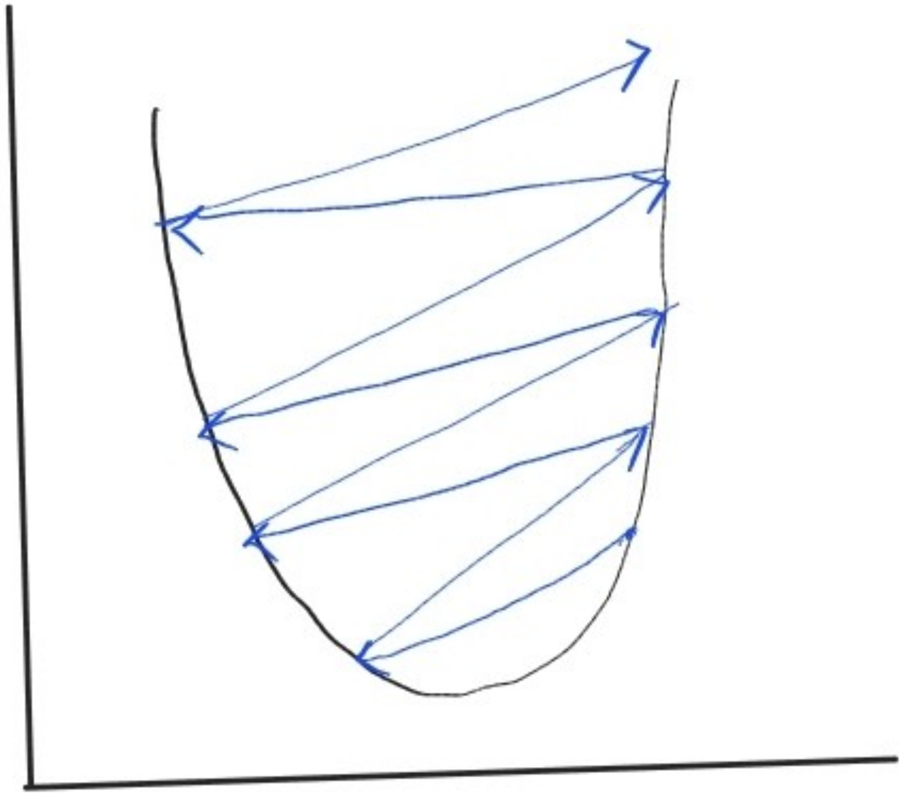
$$\begin{aligned}
\frac{\partial J}{\partial \theta_j} &= \frac{\partial}{\partial \theta_j} \frac{1}{2m} \sum (\hat{y}_i - y_i)^2 \\
&= 2 \cdot \frac{1}{2m} \sum (\hat{y}_i - y_i) \frac{\partial}{\partial \theta_j} (\hat{y}_i - y) \\
&= \frac{1}{m} \sum (\hat{y}_i - y) x_{ij} \\
\frac{\partial J}{\partial \theta_0} &= \frac{1}{m} \sum (\hat{y}_i - y)
\end{aligned}$$

- Repeat until converge

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum (\hat{y}_i - y)$$
$$\theta_j := \theta_1 - \alpha \frac{1}{m} \sum (\hat{y}_i - y) x_{ij}$$

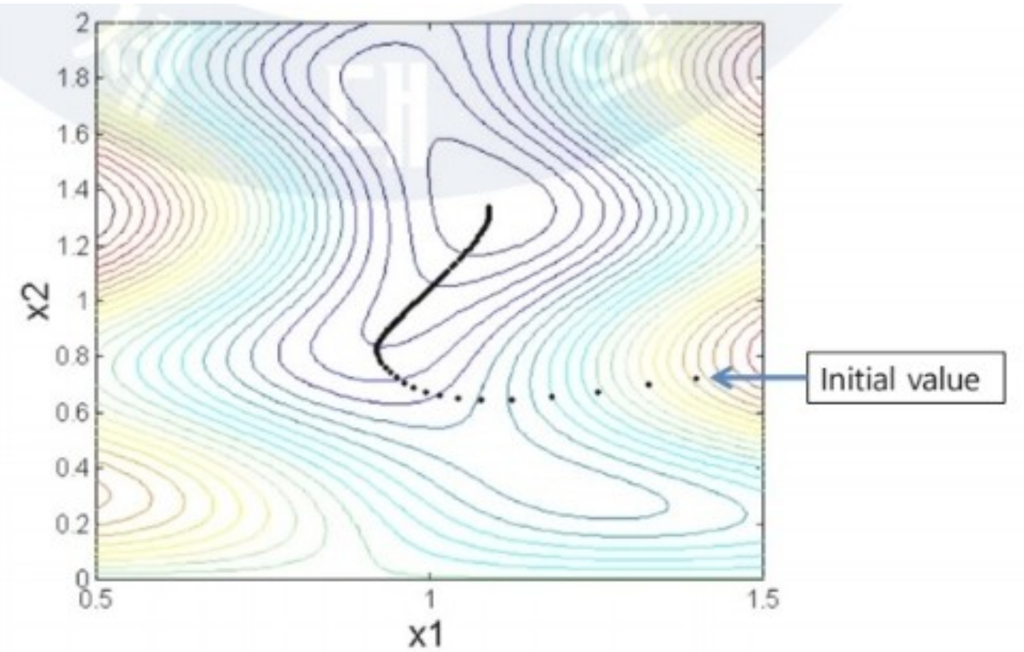
Learning rate

- If α is too high : It can diverge
- If α is too low : It is very slow to be converge



Steepest Descent

- Disadvantage of GD : converge is very slow



- If direction is not changed, there is no need to slowly update parameters

  $\rightarrow$ recalculate learning rate every step!!

- Algorithm
  1. Set $t = 0$

2. Compute $\nabla J(\theta)$
3. Compute learning rate$\alpha_i = \underset{\alpha \geq 0}{\operatorname{argmin}} f(x_i - \alpha \nabla J(\theta))$
4. Set $\theta^{(t+1)} := \theta^{(t)} - \alpha \nabla J(\theta)$
5. Repeat 2~4 until $\nabla J(\theta) = 0$