Gradient Descent Method

Data Intelligence and Learning (DIAL) Lab
Prof. Jongwuk Lee

Gradient Descent Method

Solving the Error Function

- How to **minimize the error** between $f(\mathbf{x})$ and y?

$$E(\mathbf{w}) = \frac{1}{n} \sum_{(\mathbf{x},y)\in\mathcal{D}} \big(y - f(\mathbf{x};\mathbf{w})\big)^2$$

- $E(\mathbf{w})$ is **minimum** where **the derivative of $E(\mathbf{w})$ is zero.**

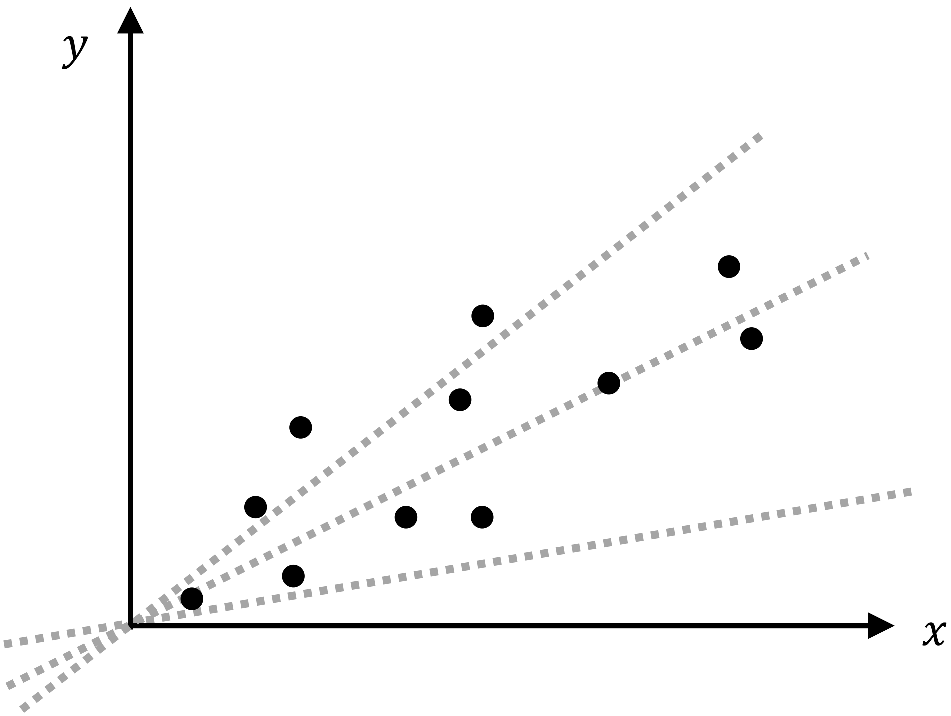
$$\left.\begin{aligned} \frac{\partial E}{\partial w_0} &= 0 \\ \frac{\partial E}{\partial w_1} &= 0 \\ &\ \ \dots \\ \frac{\partial E}{\partial w_d} &= 0 \end{aligned}\right\}$$

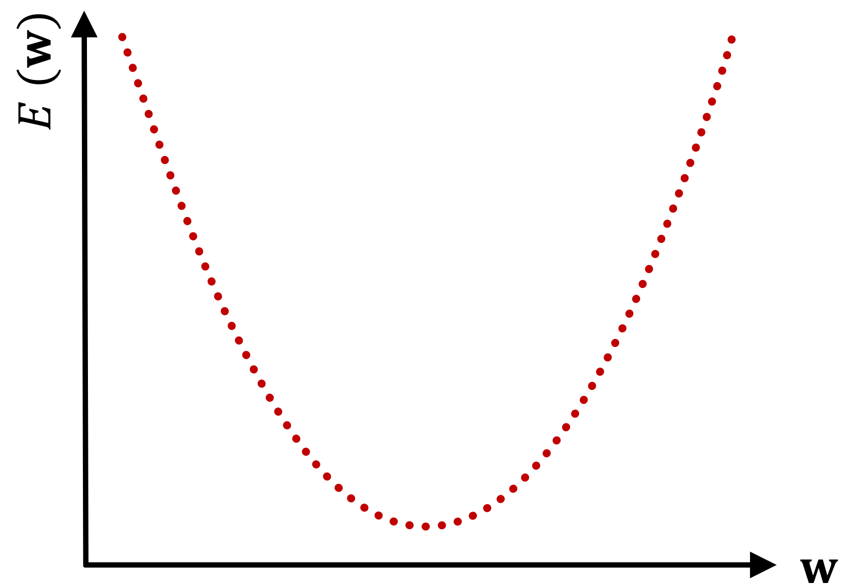
Finding $\mathbf{w}$ satisfying these equations

Solving the Error Function

- Consider a simple linear regression with w_0 and w_1.

$$\underset{w_0,w_1\in[-\infty,\infty]}{\operatorname{argmin}} E(w_0,w_1)=\frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)}-\left(w_1x^{(i)}+w_0\right)\right)^2$$





Solving the Error Function

- Does the solution always exist?
- Yes, because the error function is quadratic.

$$E(\mathbf{w}) = \frac{1}{n} \sum_{(\mathbf{x},y)\in\mathcal{D}} \left(y - f(\mathbf{x};\mathbf{w})\right)^2$$

- Can we solve it if we choose more complex models?
- No, we may not.

- What should we do if we cannot solve it?

Solving the Optimization Problem

- **The ML models can be trained analytically (e.g., normal equation) or are solved numerically (e.g., gradient descent).**

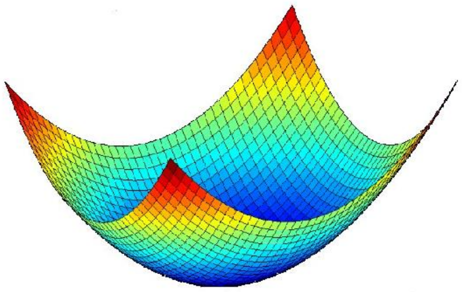
- **Analytical solution**
  - It involves framing the problem in a **well-understood form** and calculating the **exact solution**.
  - In general, it is preferred because it is **faster,** and the solution is **exact**.

- **Numerical solution**
  - **Make guesses for the solution**.
  - **It is necessary to validate whether it is solved well or not**.

https://machinelearningmastery.com/analytical-vs-numerical-solutions-in-machine-learning/

Challenge of Optimization Problems

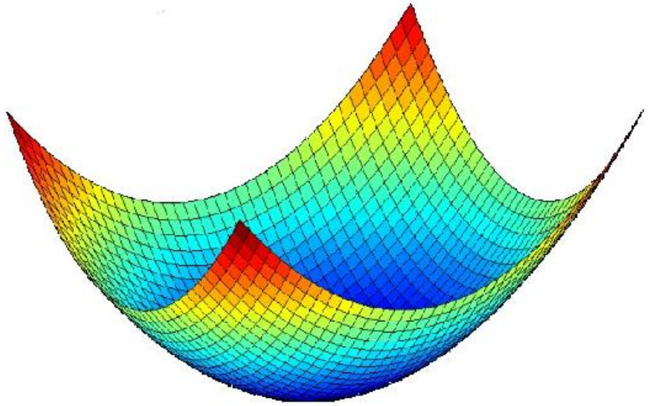
- **Finding the minimum value by the closed-form solution is often difficult or impossible.**
  - **Quadratic functions in many variables**
    - System of equations for partial derivatives may be **ill-conditioned**.
  - **Other convex functions**
    - Global minimum exists, but there is **no closed-form solution**.
    - E.g., logistic regression



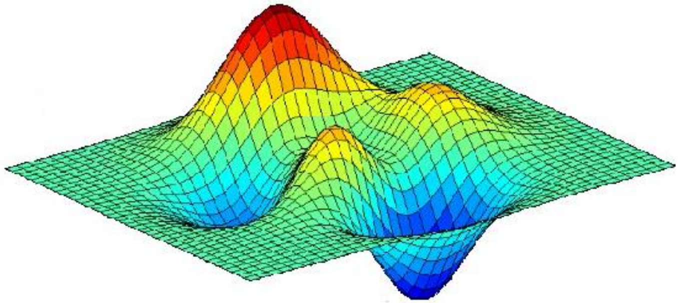
  - **Nonlinear functions**
    - Partial derivatives are **not linear**.
    - E.g., $f(x_1, x_2) = x_1(\sin(x_1 x_2)) + x_2^2$
    - E.g., sum of transfer functions in neural networks

Solutions for Optimization Problems

- **Several approximation methods**
  - **Gradient descent method**
  - Newton method
  - Gauss-Newton
  - Levenberg-Marquardt
  - BFGS
  - Conjugate gradient



Convex



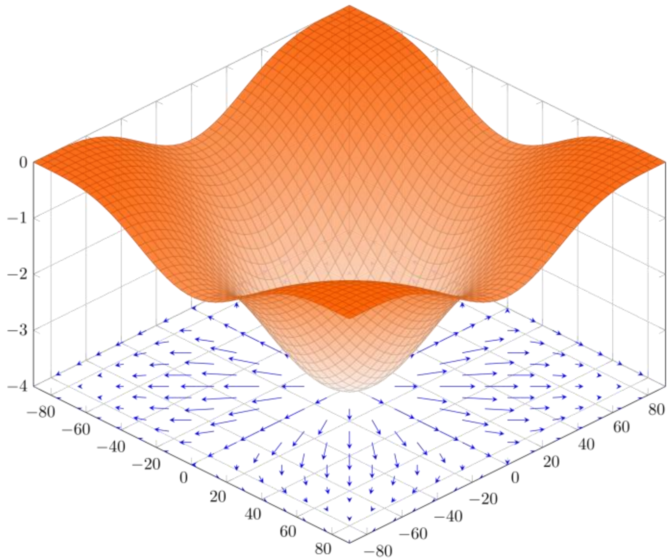
Non-convex

Intuition of Gradient Descent Method

What is the Gradient?

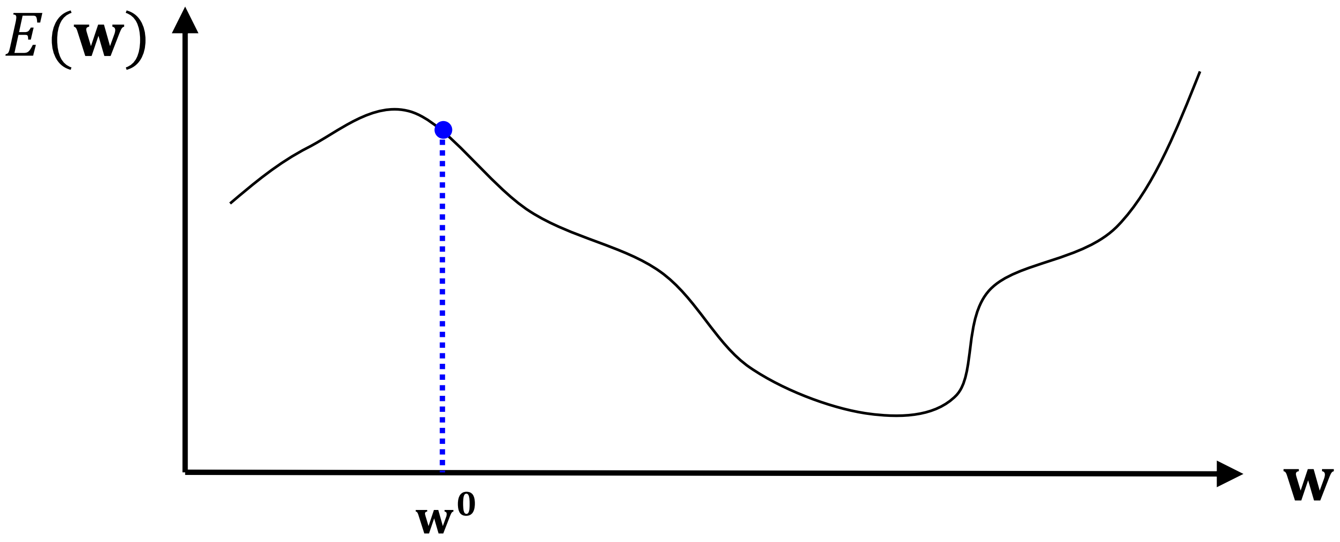
- **The gradient of a function f is the collection of all its partial derivatives into a vector.**
  - It is denoted as $\nabla f(\mathbf{w})$.
  - Each element in the gradient is the **slope of the function** along the direction of one of the variables.

$$\nabla f(\mathbf{w}) = \nabla f(w_1, w_2, \dots, w_d) = \begin{bmatrix} \frac{\partial f(\mathbf{w})}{\partial w_1} \\ \frac{\partial f(\mathbf{w})}{\partial w_2} \\ \vdots \\ \frac{\partial f(\mathbf{w})}{\partial w_d} \end{bmatrix}$$



How to Update w?

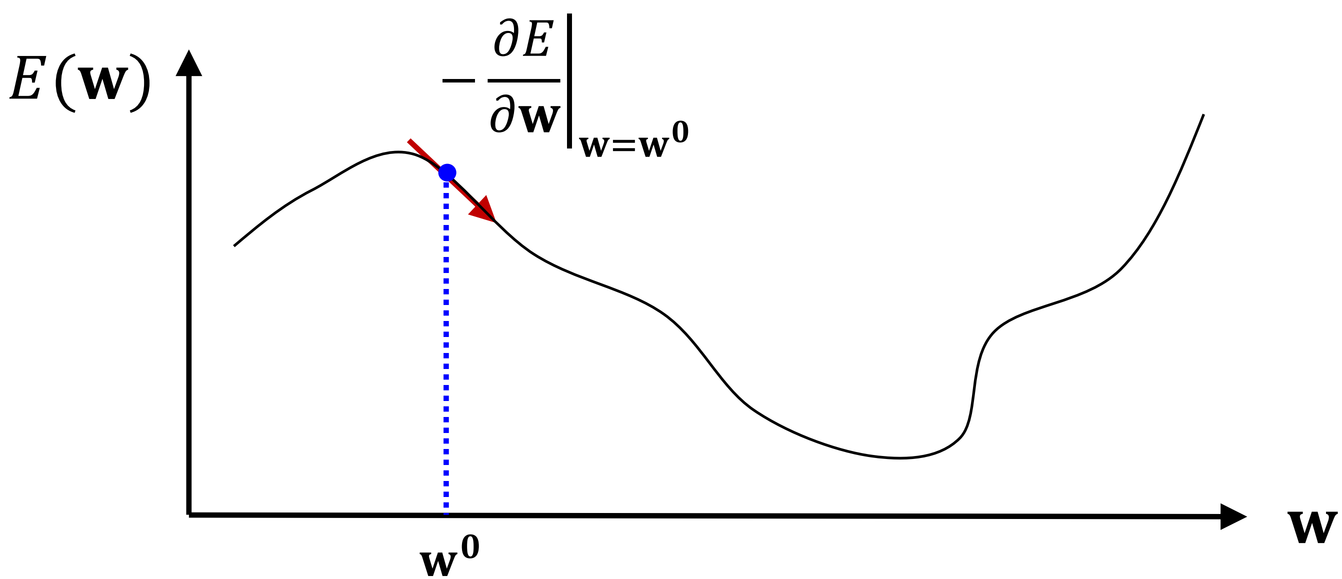
- Randomly choose **an initial point $\mathbf{w}^0$**.



How to Update w?

- **We want to update w satisfying $E(\mathbf{w_{new}}) < E(\mathbf{w})$.**
  - When $\mathbf{w_{new}} = \mathbf{w} + \epsilon$, $E(\mathbf{w_{new}}) < E(\mathbf{w})$

- **The slope at a position == differential value at a position**
  - If $E(\mathbf{w})$ is differentiable, it is easy to find out the slope.

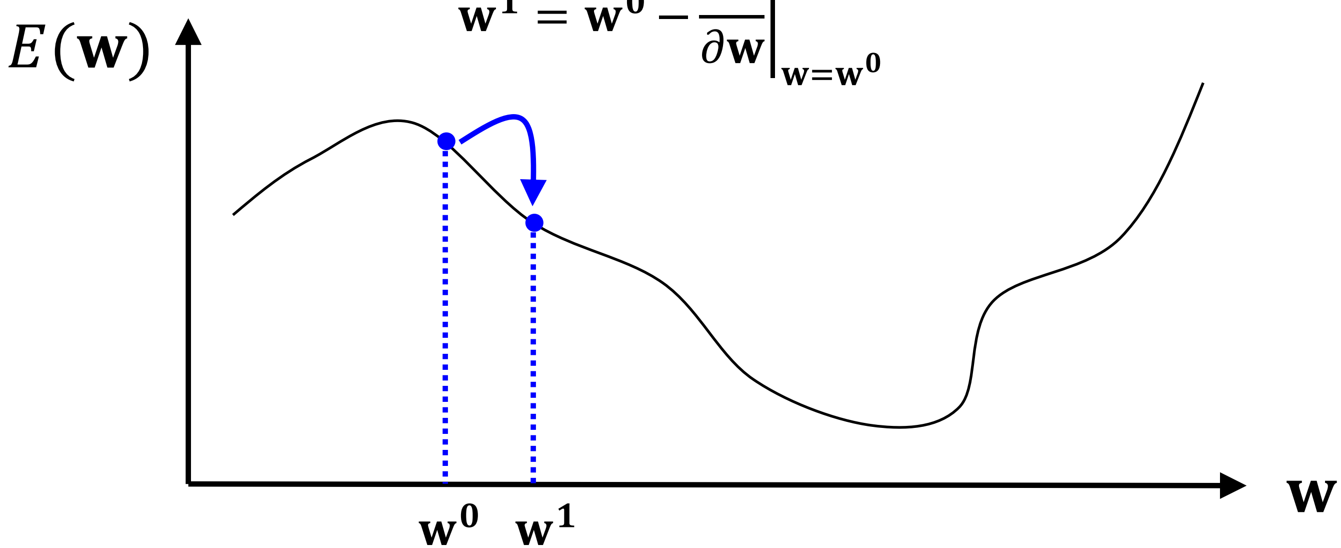


How to Update $\mathbf{w}$?

- **Move the reverse direction of the gradient.**
  - If the slope is **positive**, move $\mathbf{w}$ to the **negative** direction.
  - If the slope is **negative**, move $\mathbf{w}$ to the **positive** direction.

- **When** $\mathbf{w}_{\mathbf{new}} = \mathbf{w} - \frac{\partial E}{\partial \mathbf{w}}$, $E(\mathbf{w}_{\mathbf{new}}) < E(\mathbf{w})$

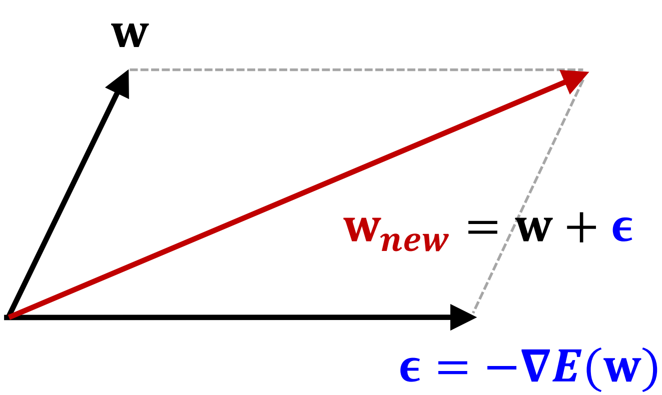
$$\mathbf{w}^1 = \mathbf{w}^0 - \left.\frac{\partial E}{\partial \mathbf{w}}\right|_{\mathbf{w}=\mathbf{w}^0}$$



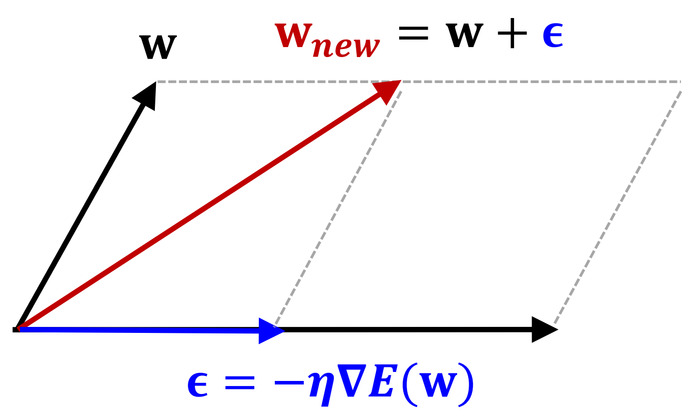
https://towardsdatascience.com/mathematical-intuition-behind-gradient-descent-f1b959a59e6d

How to Update $\mathbf{w}$?

- **Suppose that $\mathbf{w}$ is a vector and so is ϵ.**
- **When the sum of two vectors can be quite large, we need to control the size of ϵ.**
  - This is important because if we make a large update ϵ to $\mathbf{w}$, we **might miss out the global minimum** of error function $E(\mathbf{w})$.



Without the learning rate



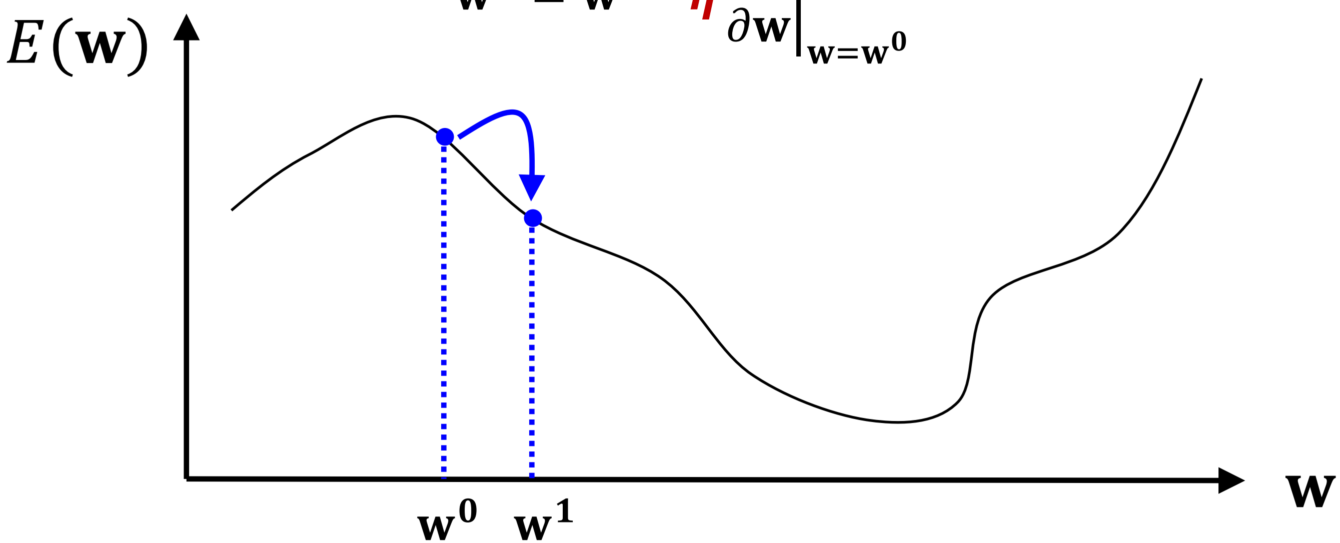
With the learning rate

How to Update w?

- Using a learning rate that limits the size of update for w

learning rate:
controlling the step size

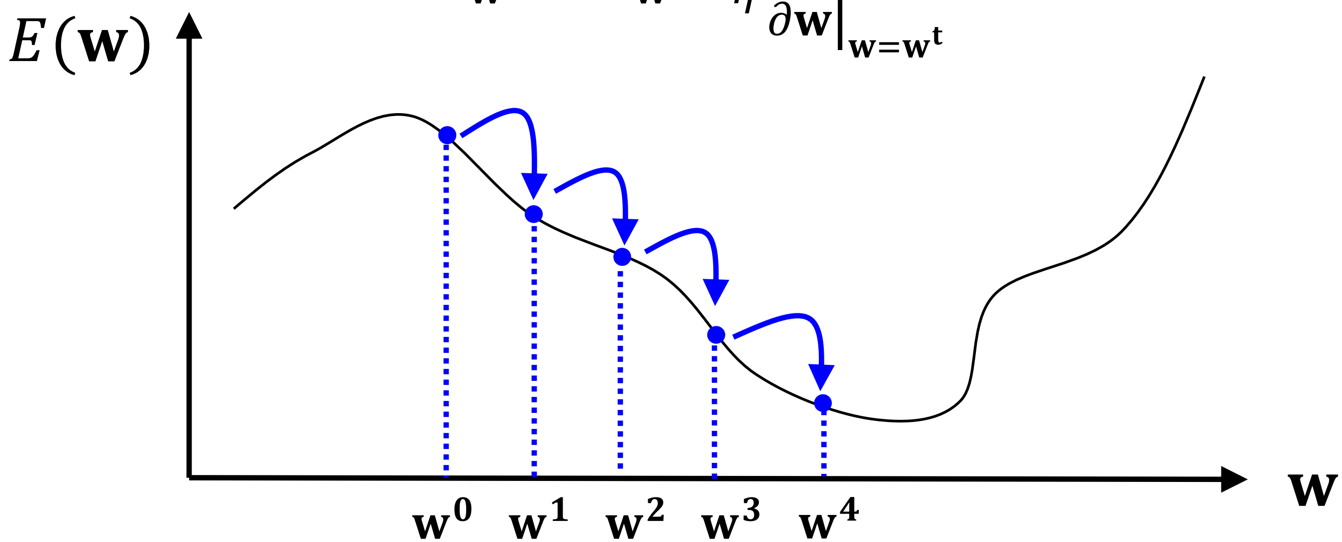
$$\mathbf{w}^1 = \mathbf{w}^0 - \eta \frac{\partial E}{\partial \mathbf{w}}\bigg|_{\mathbf{w}=\mathbf{w}^0}$$



How to Update w?

- Move until **the gradient of $E(\mathbf{w})$ is zero.**

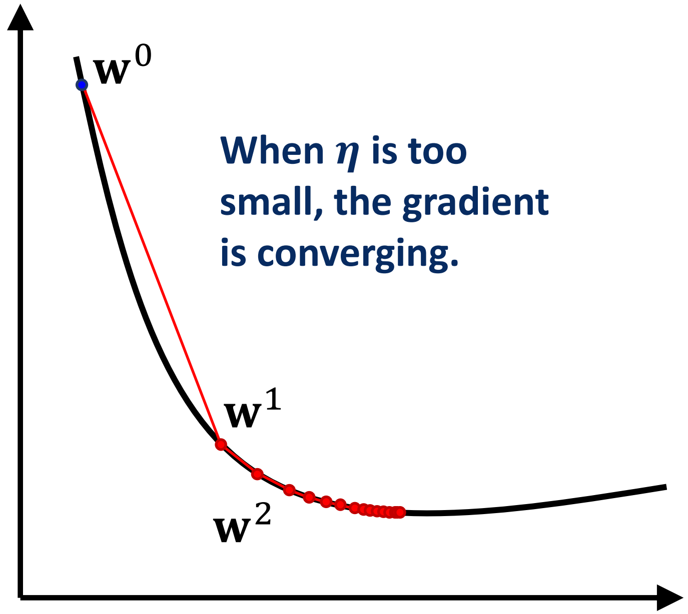
$$\mathbf{w}^{\mathbf{t}+\mathbf{1}} = \mathbf{w}^{\mathbf{t}} - \eta \left.\frac{\partial E}{\partial \mathbf{w}}\right|_{\mathbf{w}=\mathbf{w}^{\mathbf{t}}}$$

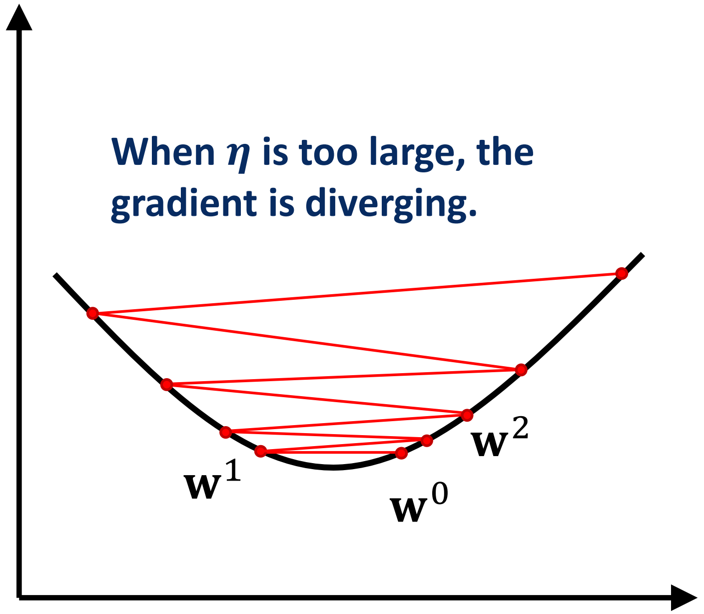


What is the Learning Rate?

- η is used to control the **step size** or **step length**.

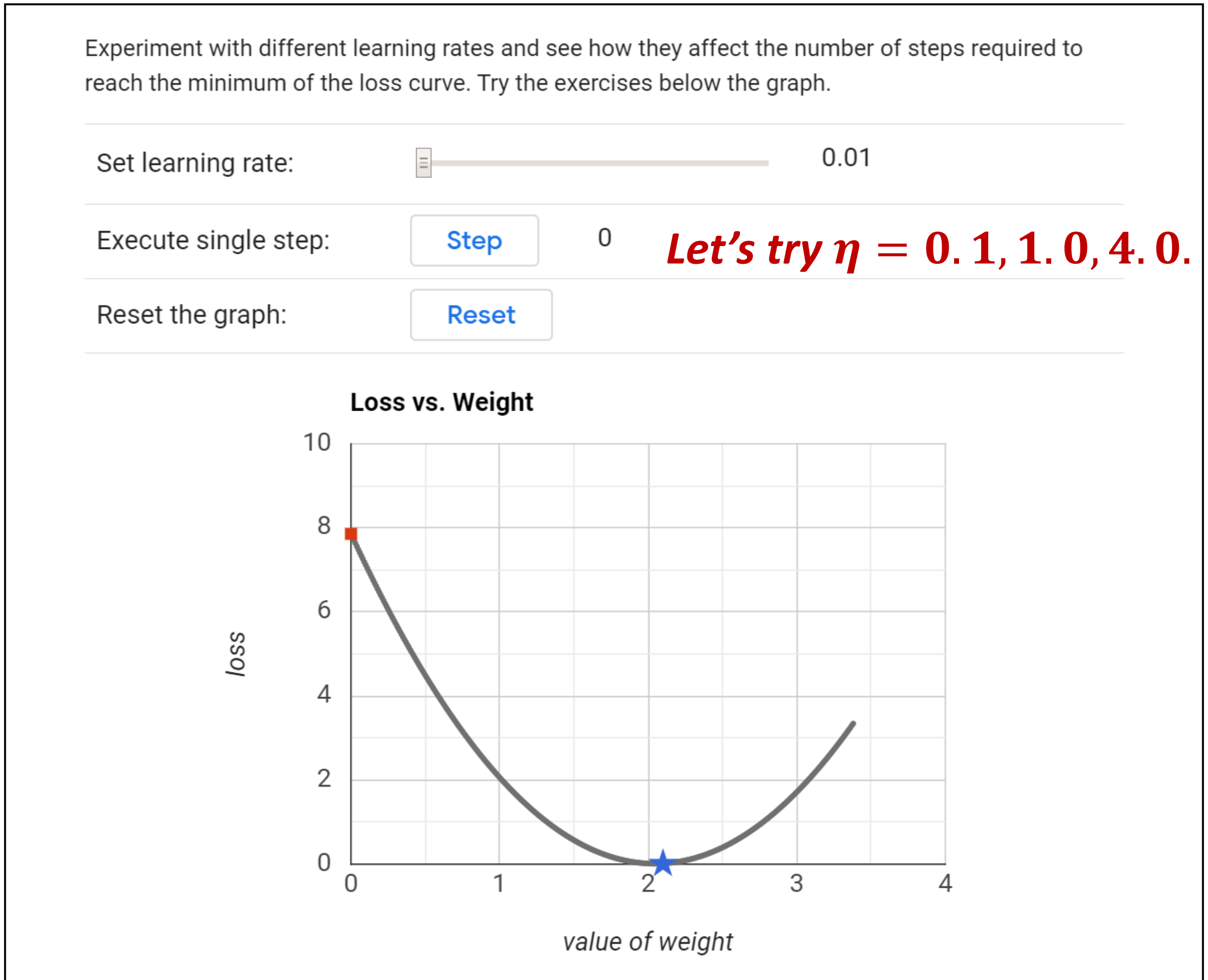
- Too small η can incur **slow convergence**.
- Too large η can incur **overshoot** the minima and **diverge**.





Example: Choosing the Learning Rate

- **Demo:** https://developers.google.com/machine-learning/crash-course/fitter/graph



Let's try $\boldsymbol{\eta = 0.1, 1.0, 4.0.}$

Gradient Descent (GD)

learning rate:
controlling the step size

Randomly choose an initial solution $\mathbf{w}^0$,

Repeat

$$\mathbf{w}^{\mathbf{t+1}} = \mathbf{w}^{\mathbf{t}} - \eta \left.\frac{dE}{d\mathbf{w}}\right|_{\mathbf{w}=\mathbf{w}^{\mathbf{t}}}$$

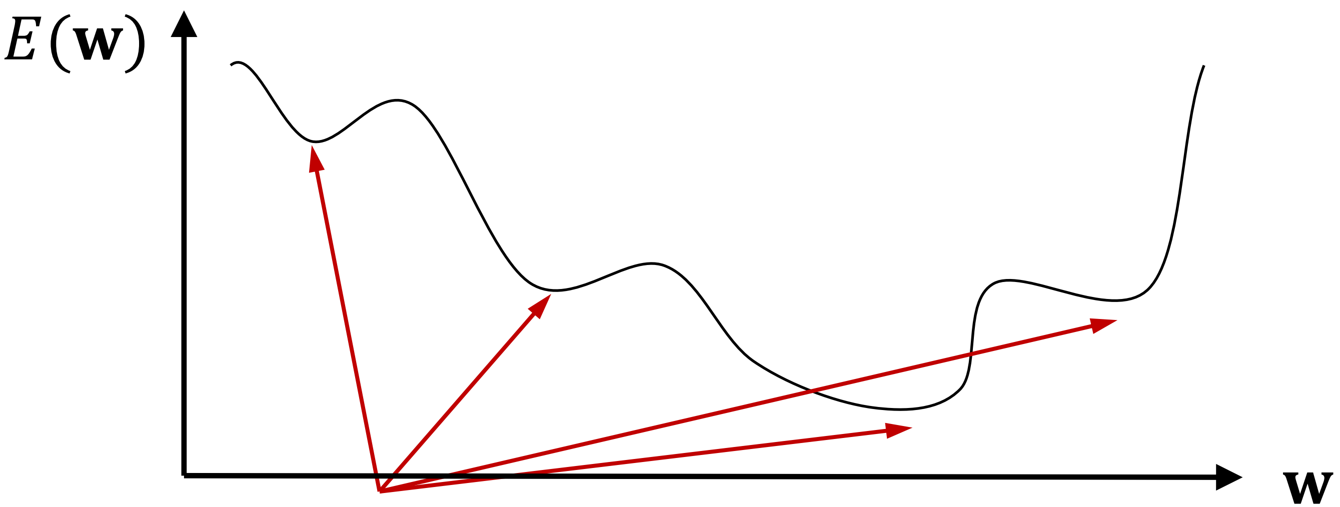
Until the stopping condition is satisfied

- Fixed number of iterations
- $|E(\mathbf{w}^{\mathbf{t+1}}) - E(\mathbf{w}^{\mathbf{t}})|$ is very small.

Finding Some Minimal Positions

- How to **minimize the error** between $f(\mathbf{x})$ and y?

- Depending on $\mathbf{w}^0$, it can find different minimums.



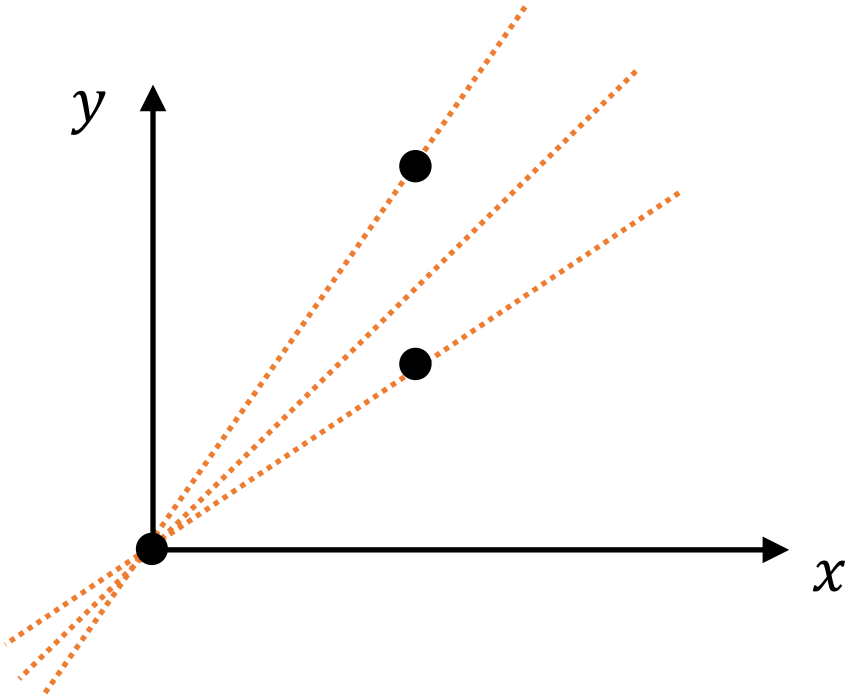
Find out any of those minimums

Example: Gradient Descent Method

Simple Linear Model

- Finding a **linear model** that fits a given data

$$f(x; w_0, w_1) = w_1 x + w_0$$



$$Data = \{(0.0, 0.0), (1.0, 1.0), (1.0, 2.0)\}$$

Solving the Optimization Problem

- **Finding w** $= (w_0, w_1)$ **that minimize** $E(w_0, w_1)$
  - For simplicity, we use sum instead of mean.

$$E(w_0, w_1) = \sum_{(x,y)\in\mathcal{D}} \left(y - f(x; w_0, w_1)\right)^2$$

$$f(x; w_0, w_1) = w_1 x + w_0$$

$$Data = \{(0.0, 0.0), (1.0, 1.0), (1.0, 2.0)\}$$

$$E(w_0, w_1) = \left(0.0 - f(0.0; w_0, w_1)\right)^2 + \left(1.0 - f(1.0; w_0, w_1)\right)^2 + \left(2.0 - f(1.0; w_0, w_1)\right)^2$$

$$E(w_0, w_1) = (0.0 - w_0)^2 + (1.0 - (w_0 + w_1))^2 + (2.0 - (w_0 + w_1))^2$$

$$E(w_0, w_1) = 2w_1^2 + 3w_0^2 - 6w_1 - 6w_0 + 4w_1 w_0 + 5$$

Calculating the Gradient

➢ **Calculate the gradient for a given error function.**

$$E(w_0, w_1) = 2w_1^2 + 3w_0^2 - 6w_1 - 6w_0 + 4w_1 w_0 + 5$$

$$\frac{\partial E}{\partial w_1} = 4w_1 + 4w_0 - 6$$

$$\frac{\partial E}{\partial w_0} = 4w_1 + 6w_0 - 6$$

Randomly choose an initial solution, $\boldsymbol{w_0^0}$, $\boldsymbol{w_1^0}$.

Repeat

$$w_0^{t+1} = w_0^t - \eta \left.\frac{\partial E}{\partial w_0}\right|_{w_0 = w_0^t, w_1 = w_1^t}$$

$$w_1^{t+1} = w_1^t - \eta \left.\frac{\partial E}{\partial w_1}\right|_{w_0 = w_0^t, w_1 = w_1^t}$$

Until the stopping condition is satisfied

Calculating the Gradient

- **Find w_0, w_1 that minimizes the error function.**
  - Choose a learning rate η, e.g., $\eta = 0.1$.
  - Initialize w_0^0, w_1^0 as random values, e.g., $w_0^0 = 1, w_1^0 = 1$

Randomly choose an initial solution, $\boldsymbol{w_0^0}$, $\boldsymbol{w_1^0}$.

Repeat

$$w_0^{t+1} = w_0^t - \eta(4w_1^t + 6w_0^t - 6)$$

$$w_1^{t+1} = w_1^t - \eta(4w_1^t + 4w_0^t - 6)$$

Until the stopping condition is satisfied

How does the GD Work?

- **Update w_0^t, w_1^t iteratively.**

Randomly choose an initial solution, $\boldsymbol{w_0^0}$, $\boldsymbol{w_1^0}$.

Repeat

$$w_0^{t+1} = w_0^t - \eta(4w_1^t + 6w_0^t - 6)$$

$$w_1^{t+1} = w_1^t - \eta(4w_1^t + 4w_0^t - 6)$$

Until the stopping condition is satisfied

$$w_0^0 = 1$$
$$w_1^0 = 1$$

$$w_0^1 = 1 - 0.1(4 \times 1 + 6 \times 1 - 6) = 0.6$$
$$w_1^1 = 1 - 0.1(4 \times 1 + 4 \times 1 - 6) = 0.8$$

$$w_0^2 = 0.6 - 0.1(4 \times 0.8 + 6 \times 0.6 - 6) = 0.54$$
$$w_1^2 = 0.8 - 0.1(4 \times 0.8 + 4 \times 0.6 - 6) = 0.84$$

$$w_0^3 = 0.54 - 0.1(4 \times 0.84 + 6 \times 0.54 - 6) = 0.480$$
$$w_1^3 = 0.84 - 0.1(4 \times 0.84 + 4 \times 0.54 - 6) = 0.888$$

How does the GD Work?

- Update w_0^t, w_1^t iteratively.

Randomly choose an initial solution, $\boldsymbol{w_0^0}, \boldsymbol{w_1^0}$.

Repeat

$$w_0^{t+1} = w_0^t - \eta(4w_1^t + 6w_0^t - 6)$$

$$w_1^{t+1} = w_1^t - \eta(4w_1^t + 4w_0^t - 6)$$

Until the stopping condition is satisfied

$$w_0^3 = \mathbf{0.480}$$
$$w_1^3 = \mathbf{0.888}$$

$$w_0^4 = \mathbf{0.480} - 0.1(4 \times \mathbf{0.888} + 6 \times \mathbf{0.480} - 6) = \mathbf{0.4368}$$
$$w_1^4 = \mathbf{0.888} - 0.1(4 \times \mathbf{0.888} + 4 \times \mathbf{0.480} - 6) = \mathbf{0.9408}$$

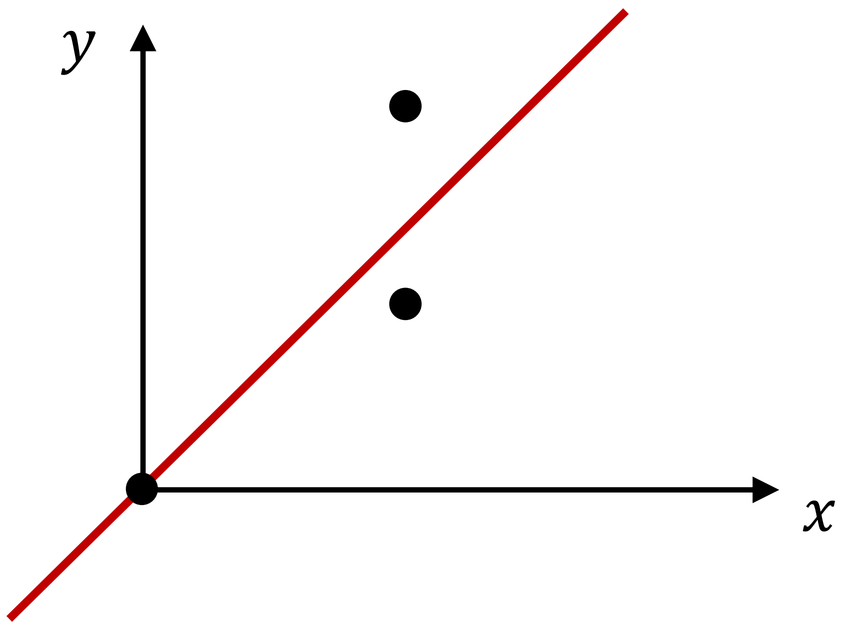
…

$$w_0^{100} = \mathbf{0.00007713}$$
$$w_1^{100} = \mathbf{1.49989171}$$

Solution of the Linear Model

- **Finding a linear model that fits a given data**

$$f(x; w_0, w_1) = 1.49989171x + 0.00007713$$



$$Data = \{(0.0, 0.0), (1.0, 1.0), (1.0, 2.0)\}$$

Details: Gradient Descent Method

Taylor Series Expansion

- **Taylor series function is an infinite sum of terms that are expressed by the function's derivatives at a single point.**
  - $f^{(k)}(a)$: k-th derivative of the function

$$f(x) = f(a) + \frac{f^{(1)}(a)}{1!}(x-a) + \frac{f^{(2)}(a)}{2!}(x-a)^2 + \cdots + \frac{f^{(k)}(a)}{k!}(x-a)^k + \cdots$$

$$\Downarrow$$

$$f(x) = \sum_{i=0}^{\infty} \frac{f^{(i)}(a)}{i!}(x-a)^i$$

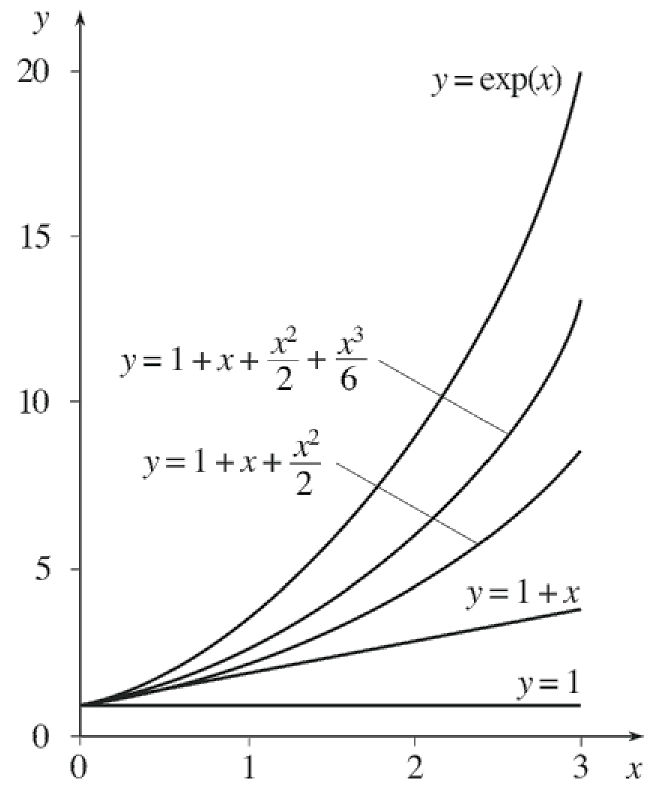
Example: Taylor Series Expansion

- $f(x) = e^x$ and $a = 0$

$$e^x = f(0) + \frac{f^{(1)}(0)}{1!}(x-0) + \frac{f^{(2)}(0)}{2!}(x-0)^2 + \cdots + \frac{f^{(k)}(0)}{k!}(x-0)^k + \cdots$$

$$\Downarrow$$

$$e^x = 1 + x + \frac{x^2}{2!} + \cdots + \frac{x^k}{k!} + \cdots$$



https://www.desmos.com/calculator/vfhuujhdqg?lang=ko

Taylor Series Expansion for $f(x+\epsilon)$

- Suppose that $x = x_1 + \epsilon$ and $a = x_1$

$$f(x) = f(a) + \frac{f^{(1)}(a)}{1!}(x-a) + \frac{f^{(2)}(a)}{2!}(x-a)^2 + \cdots + \frac{f^{(k)}(a)}{k!}(x-a)^k + \cdots$$

$$(x-a) = (x_1 + \epsilon - x_1) = \epsilon$$

$$f(x_1+\epsilon) = f(x_1) + \frac{f^{(1)}(x_1)}{1!}\epsilon + \underbrace{\frac{f^{(2)}(x_1)}{2!}\epsilon^2 + \cdots + \frac{f^{(k)}(x_1)}{k!}\epsilon^k + \cdots}$$

$$f(x_1+\epsilon) = f(x_1) + \epsilon f^{(1)}(x_1) + O(\epsilon^2)$$

The rest of the terms are restricted by $O(\epsilon^2)$.

Gradient Descent in One Dimension

- **The first-order approximation $f(x+\epsilon)$ is given by $f(x)$ and $f^{(1)}(x)$ at x.**

$$f(x+\epsilon) = f(x) + \epsilon f^{(1)}(x) + O(\epsilon^2)$$

- **We want to choose ϵ to preserve $f(x+\epsilon) \lesssim f(x)$**

$$f(x+\epsilon) - f(x) = \epsilon f^{(1)}(x) + O(\epsilon^2) \lesssim 0$$

$\Downarrow$ Assuming ϵ is too small, $O(\epsilon^2)$ is negligible.

$$f(x+\epsilon) - f(x) \approx \epsilon f^{(1)}(x) \lesssim 0$$

Gradient Descent in One Dimension

- **Pick a fixed step size $\eta > 0$ and choose $\epsilon = -\eta f^{(1)}(x)$.**
  - If $f^{(1)}(x) \neq 0$, then $\epsilon f^{(1)}(x) = -\eta \left(f^{(1)}(x)\right)^2 < 0$, where $\left(f^{(1)}(x)\right)^2$ is always positive.

- **When we choose η small enough,**

$$f(x+\epsilon) - f(x) \approx \epsilon f^{(1)}(x) \lesssim 0$$

$$\Downarrow$$

$$f\left(x - \eta f^{(1)}(x)\right) - f(x) \approx -\eta \left(f^{(1)}(x)\right)^2 \lesssim 0$$

Note that η is a small positive number.

https://d2l.ai/chapter_optimization/gd.html#gradient-descent-in-one-dimension

Gradient Descent in One Dimension

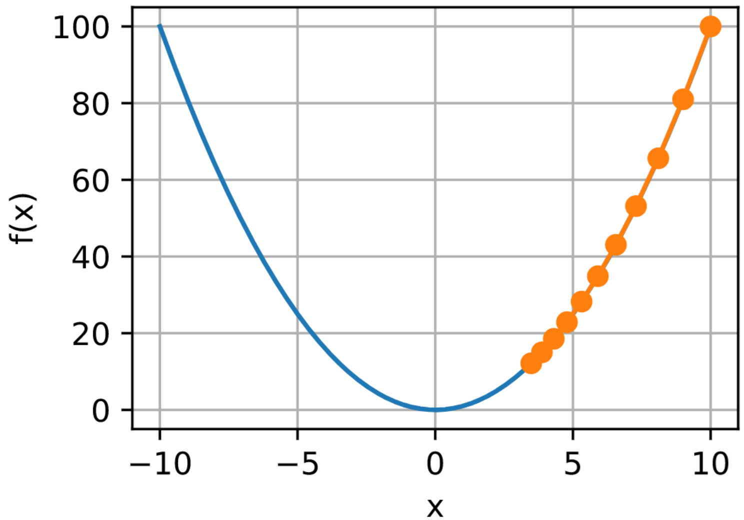
- Finally, we arrive at

$$f\left(x-\eta f^{(1)}(x)\right)-f(x)\approx-\eta\left(f^{(1)}(x)\right)^2\lesssim 0 \quad \Rightarrow \quad f\left(x-\eta f^{(1)}(x)\right)\lesssim f(x)$$

- This means that, if we use

$$x\leftarrow x-\eta f^{(1)}(x)$$

- The value of $f(x)$ might decline.



Gradient Descent in Multi-Dimension

- Let us consider the situation $\mathbf{x} \in \mathbb{R}^{d}$.

$$f(\mathbf{x} + \boldsymbol{\epsilon}) = f(\mathbf{x}) + \boldsymbol{\epsilon}^{\mathrm{T}} \nabla f(\mathbf{x}) + O(\boldsymbol{\epsilon}^{2}) \approx f(\mathbf{x}) + \boldsymbol{\epsilon}^{\mathrm{T}} \nabla f(\mathbf{x})$$

- Up to second order terms in ϵ, the direction of steepest decent is given by the negative gradient $-\nabla f(\mathbf{x})$.

- Choosing a suitable learning rate $\eta > 0$ yields the gradient descent algorithm.

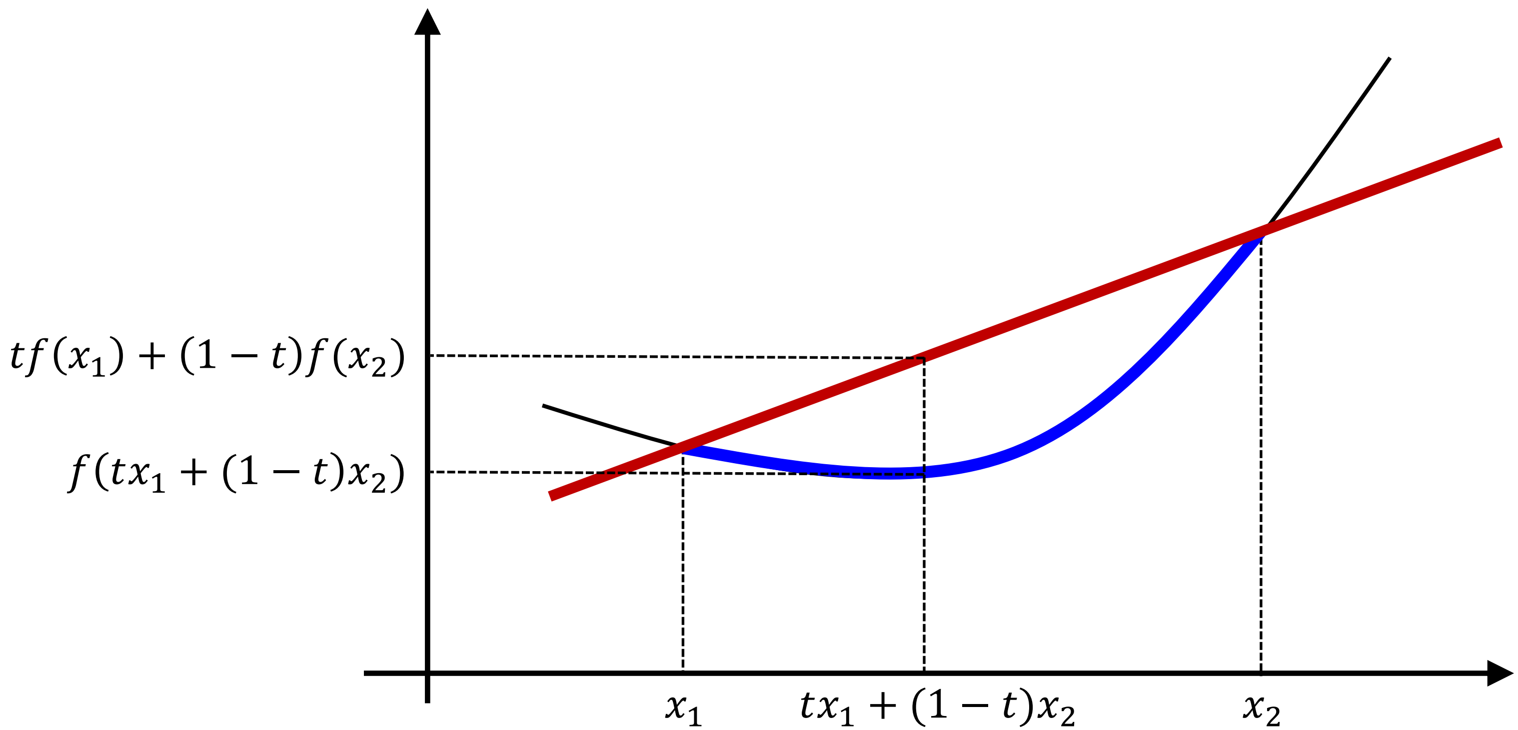
$$\mathbf{x} \leftarrow \mathbf{x} - \eta \nabla f(\mathbf{x})$$

Convex Function

What is the Convex Function?

- A function f is convex $\Leftrightarrow$ the function f is **below** any line segment between two points on f.

$f(tx + (1-t)x') \leq tf(x) + (1-t)f(x')$ for any $x, x' \in X$ and $t \in [0,1]$



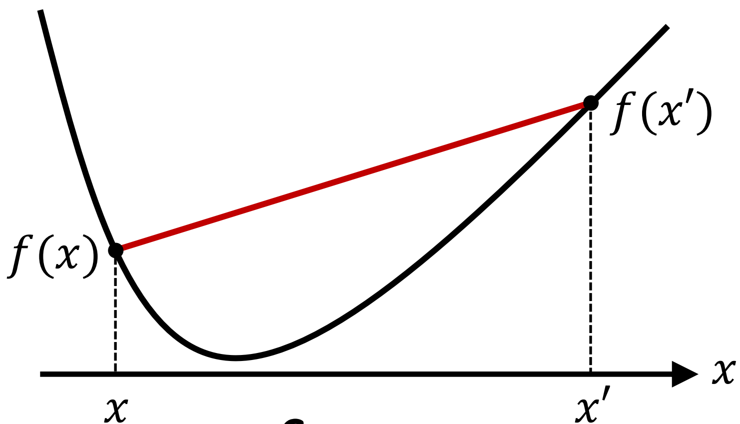
What is the Convex Function?

➢ A function $f: X \rightarrow \mathbb{R}$ is defined as a convex function if

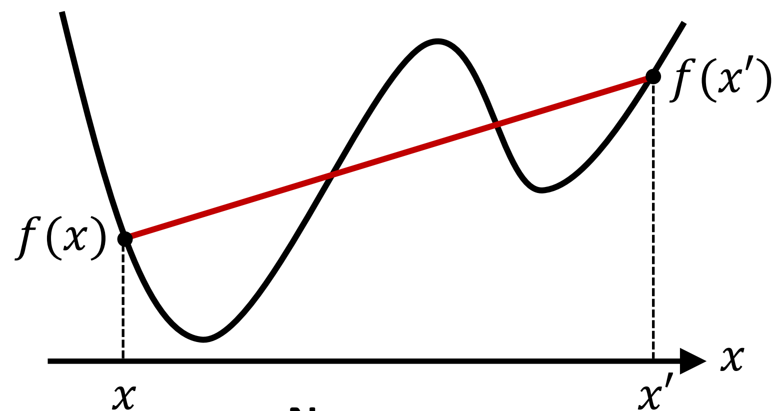
$$f(tx + (1-t)x') \leq tf(x) + (1-t)f(x') \text{ for any } x, x' \in X \text{ and } t \in [0,1]$$

$$f\left(\frac{x+x'}{2}\right) \leq \frac{f(x)+f(x')}{2} \text{ for any } x, x' \in X \text{ and } t = 0.5$$

Midpoint convex



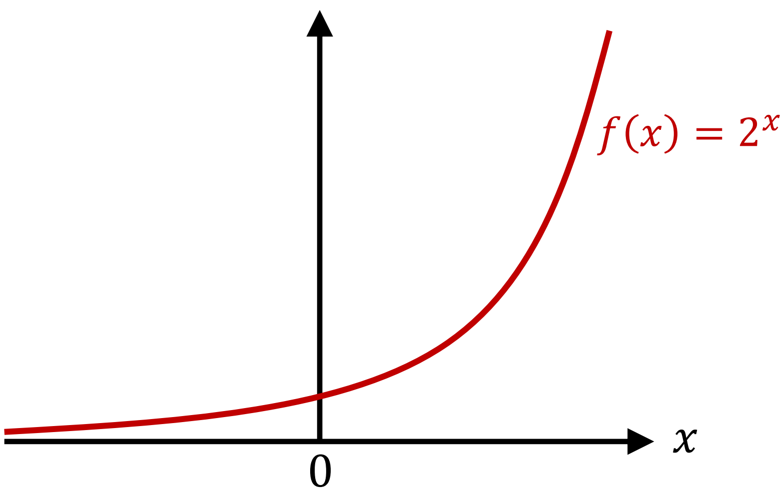
Convex

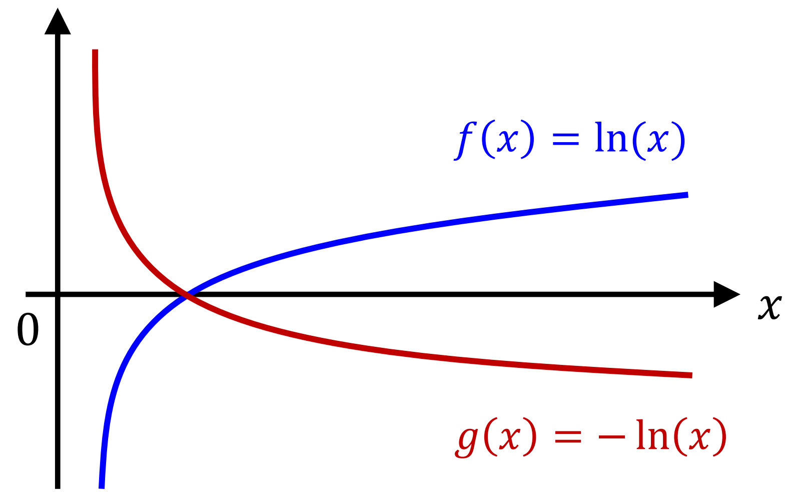


Non-convex

Examples of Convex Functions

- **Quadratic function:** $f(x) = x^2$
- **Exponential functions:** $f(x) = 2^x$
- **Negative logarithm function:** $f(x) = -\ln x$

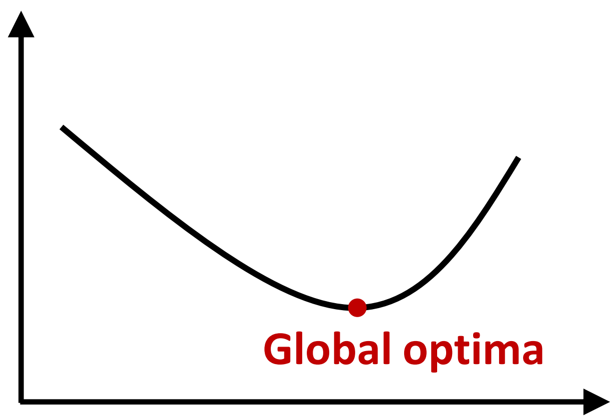




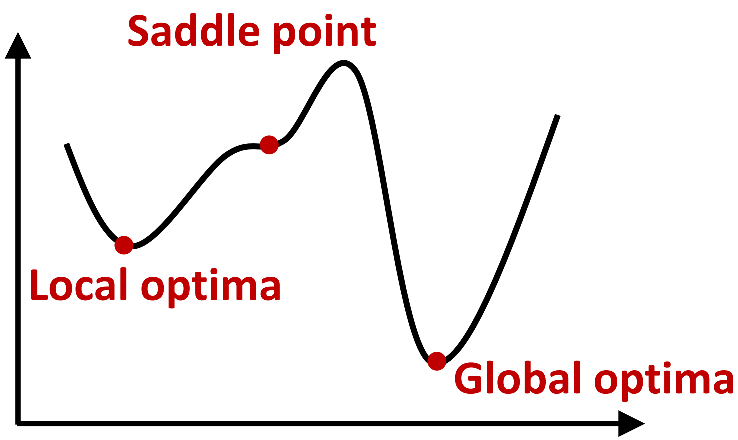
- **Convex optimization overview**
  - http://web.stanford.edu/class/cs224n/readings/cs229-cvxopt.pdf

Convex vs. Non-convex Function

- **Convex function**
  - $\nabla f(\mathbf{w}^*) = \mathbf{0} \iff \mathbf{w}^*$ **is a global minimum.**
  - Example: linear regression, logistic regression
- **Non-convex function**
  - $\nabla f(\mathbf{w}^*) = \mathbf{0} \iff \mathbf{w}^*$ **is a global min, local min or saddle point (also called stationary points.)**
  - Most algorithms only converge to stationary points.
  - Example: neural networks



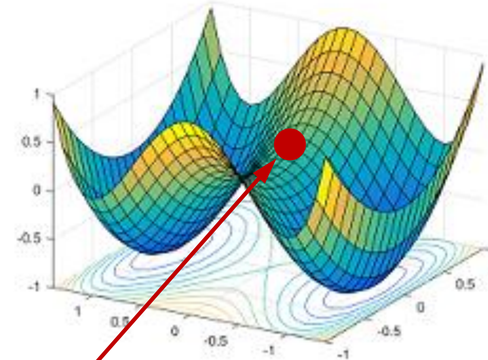
Convex



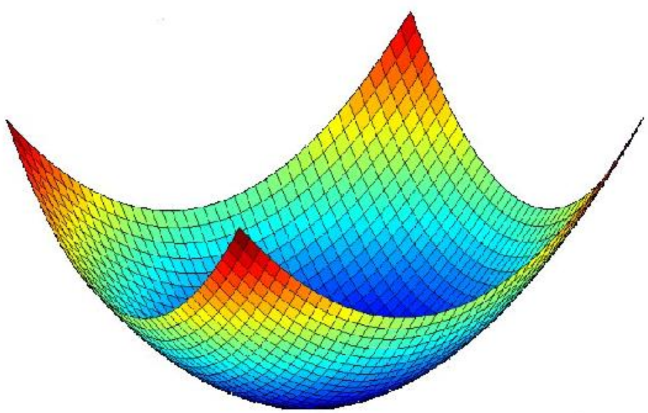
Non-convex

Convex vs. Non-convex Function

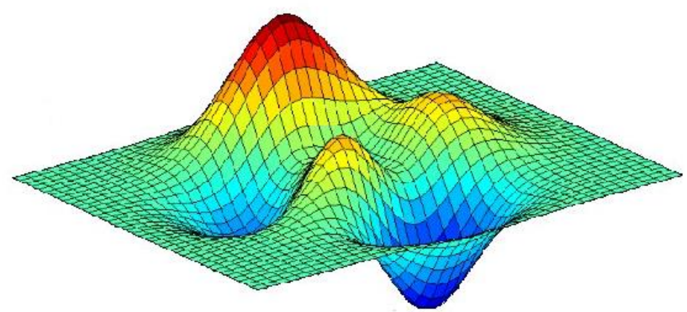
- **Given $y = x_1^4 - 2x_1^2 + x_2^2$, it has two symmetric local min $(-1,0)$ and $(1,0)$, and a saddle point $(0,0)$ between them.**



Saddle point



Convex



Non-convex

https://www.offconvex.org/2016/03/22/saddlepoints/

Q&A