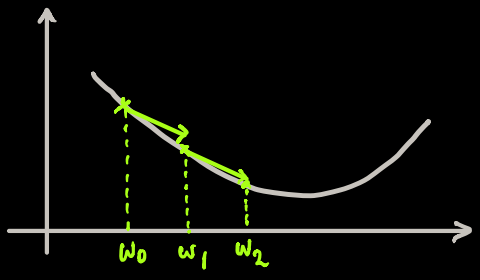
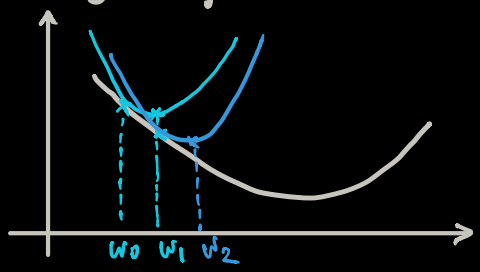
MIL 5

previously ML - Blink 4.4

- Gradient descent for loss function $\ell(w)$ optimization $[\min_w \ell(w)]$ (first order)

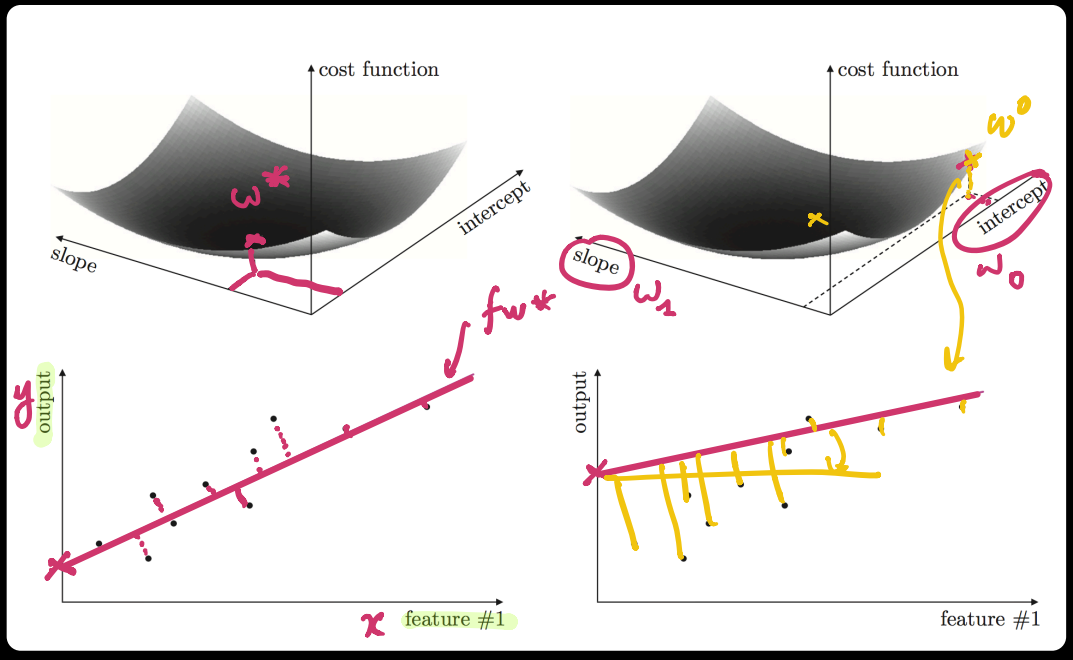


1 - Newton's method for optimization (second order)

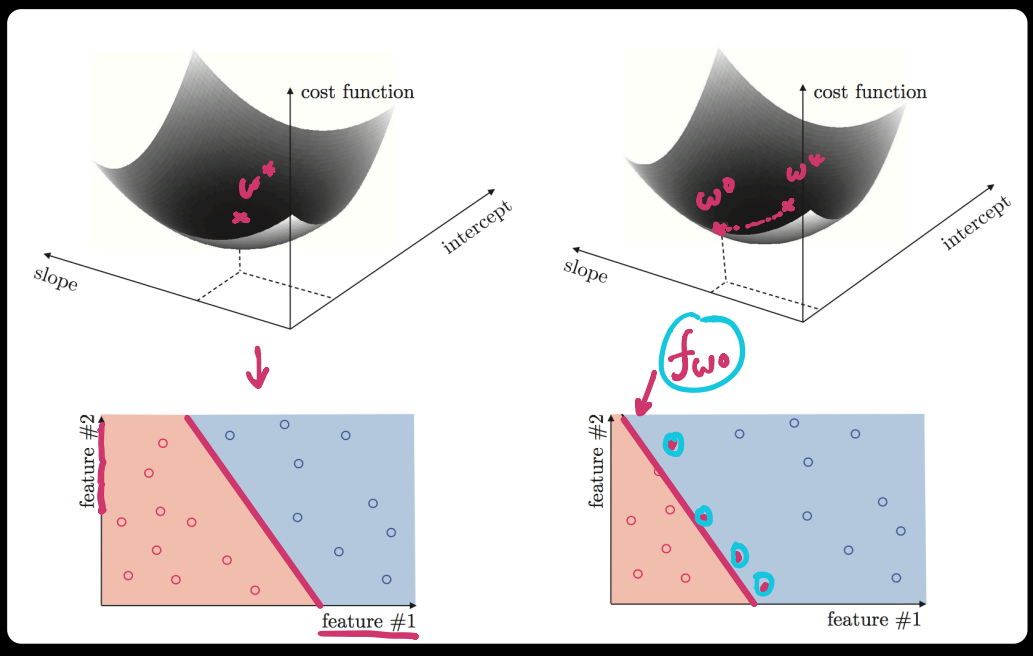


2 - Two major issues with gradient descent (zigzagging ↑ direction & vanishing ↑ magnitude)

$$arg\ min_{\mathbf{w}} \mathcal{L}(\mathbf{w}) = \frac{1}{n} \sum_{(\mathbf{x}^i, y^i) \in \mathcal{D}} E(f_{\mathbf{w}}(\mathbf{x}^i), y^i) + R(\dots)$$

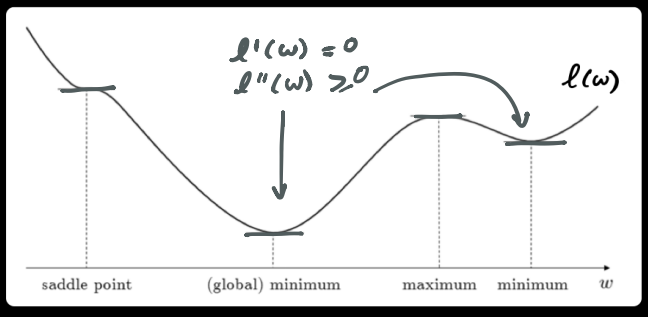


$$arg\ min_{\mathbf{w}} \overbrace{\mathcal{L}(\mathbf{w})}^{\mathcal{L}(w)} = \frac{1}{n} \sum_{(\mathbf{x}^i, y^i) \in \mathcal{D}} E(f_{\mathbf{w}}(\mathbf{x}^i), y^i) + R(\dots)$$



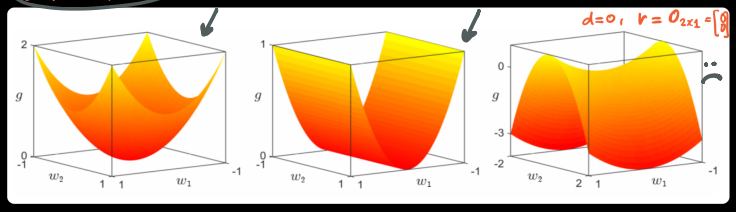
$\dot{M} \dot{\infty} \dot{L} \curvearrowright ?$

Univariate loss function
$w \in \mathbb{R}$



Multivariate loss function
$w \in \mathbb{R}^N$

$\ell(w) = w^T Q w$



$Q = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$
$\nabla^2 \ell(w) = Q$
$\lambda_1 = \lambda_2 = 2$

$Q = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}$
$\nabla^2 \ell(w) = Q$
$\begin{cases} \lambda_1 = 2 \\ \lambda_2 = 0 \end{cases}$

$Q = \begin{bmatrix} 2 & 0 \\ 0 & -2 \end{bmatrix}$
$\nabla^2 \ell(w) = Q$
$\begin{cases} \lambda_1 = 2 \\ \lambda_2 = -2 \end{cases}$

Algorithm 2.1 Gradient descent (with fixed step length)

Input: differentiable function $\boldsymbol{\ell}$, fixed step length α, and initial point $\mathbf{w}^0$

$\varepsilon = 10^{-3}$

$k = 1$

Repeat until stopping condition is met: $\|\nabla \ell(w^k)\|_2 \leq \varepsilon$

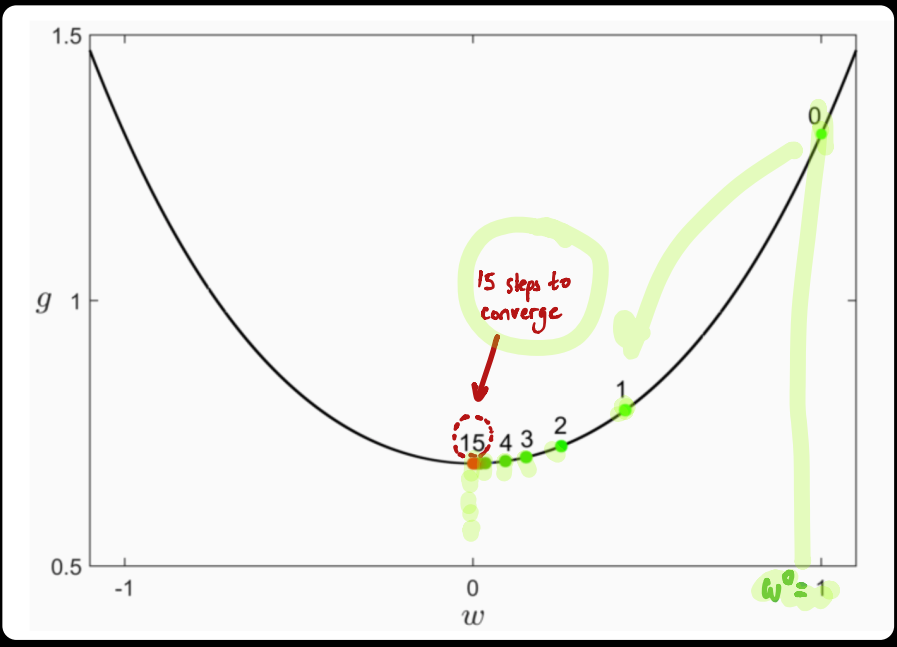
$\mathbf{w}^k = \mathbf{w}^{k-1} - \alpha \nabla \boldsymbol{\ell}(\mathbf{w}^{k-1})$

$k \leftarrow k + 1$

Example 1

$w \in \mathbb{R}$ (scalar) univariate loss function

$\ell(w) = \log(1 + e^{w^2})$ [derivative-calculator.net]



- $\ell'(w) = \dfrac{2e^{w^2} w}{1 + e^{w^2}}$
- $\alpha = 10^{-1} = 0.1$ (fixed by trial & error)
- $w^0 = 1$
- $\varepsilon = 10^{-3} = 0.001$

[derivative-calculator.net]

YOUR INPUT:
$f(x) =$

$$\ln\left(e^{x^2} + 1\right)$$

Note: Your input has been rewritten/simplified.

Simplify | Roots/zeros

FIRST DERIVATIVE:
$\frac{d}{dx}[f(x)] = f'(x) =$

The steps of calculation are displayed.
Click at any derivative $\frac{d}{dx}[\ldots]$ in order to show the rule that was applied.

$$\frac{d}{dx}\left[\ln\left(e^{x^2} + 1\right)\right]$$

$$= \frac{1}{e^{x^2} + 1} \cdot \frac{d}{dx}\left[e^{x^2} + 1\right]$$

$$= \frac{\frac{d}{dx}\left[e^{x^2}\right] + \frac{d}{dx}[1]}{e^{x^2} + 1}$$

$$= \frac{e^{x^2} \cdot \frac{d}{dx}\left[x^2\right] + 0}{e^{x^2} + 1}$$

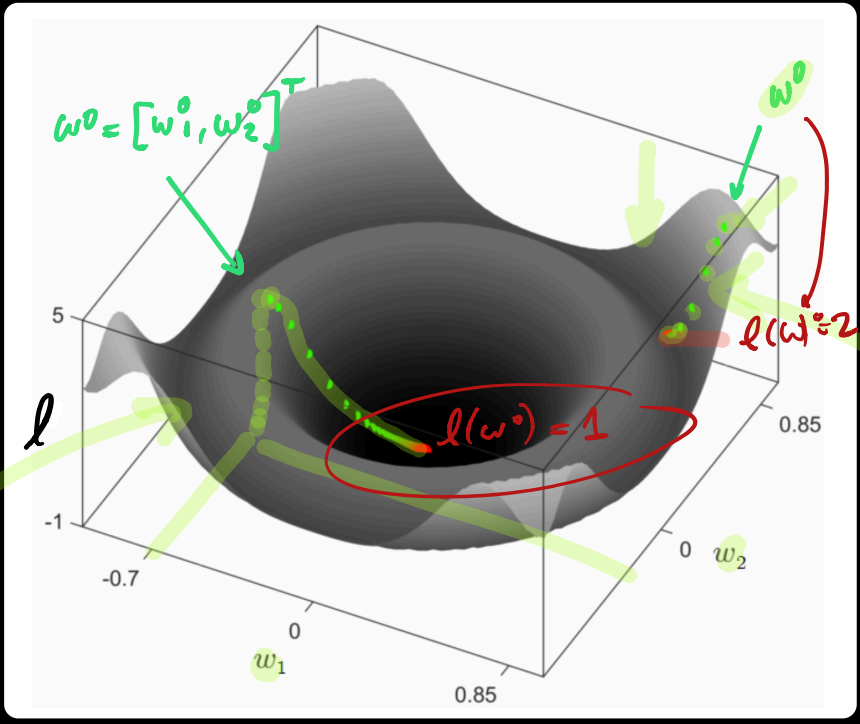
$$= \frac{2xe^{x^2}}{e^{x^2} + 1}$$

Simplify | Roots/zeros

Example 2

$$\begin{cases} w \in \mathbb{R}^2 \Rightarrow \text{multivariate loss function} \\ \ell(w) = -\cos\left(2\pi w^T w\right) + 2 w^T w \end{cases}$$

- $\nabla \ell(w) = 4\pi \sin\left(2\pi w^T w\right) w + 4w$



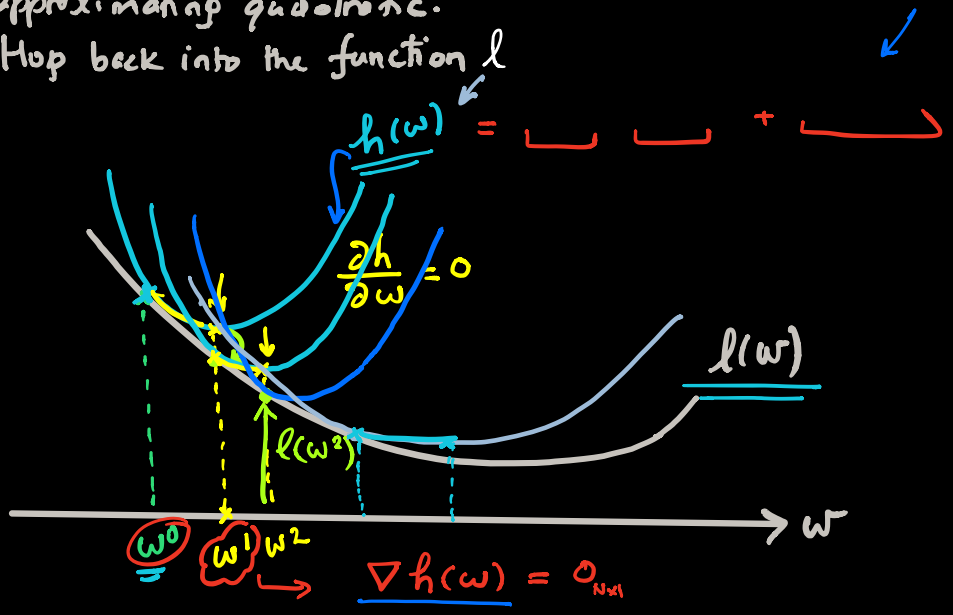
- $\alpha = 10^{-3}$
- $\varepsilon = 10^{-3}$
- $w^0 = [0.85 \quad 0.85]^T \rightarrow$ local minimum ☹
- $w^0 = [-0.7 \quad 0]^T \rightarrow$ global minimum ☺

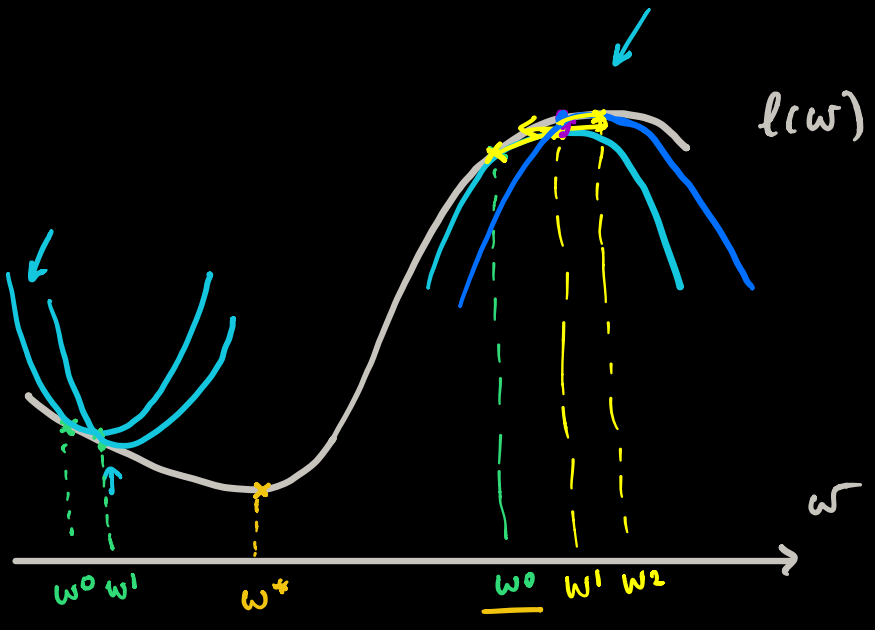
② Basic technique 2 for finding $\min_w \ell(w)$: Newton's method

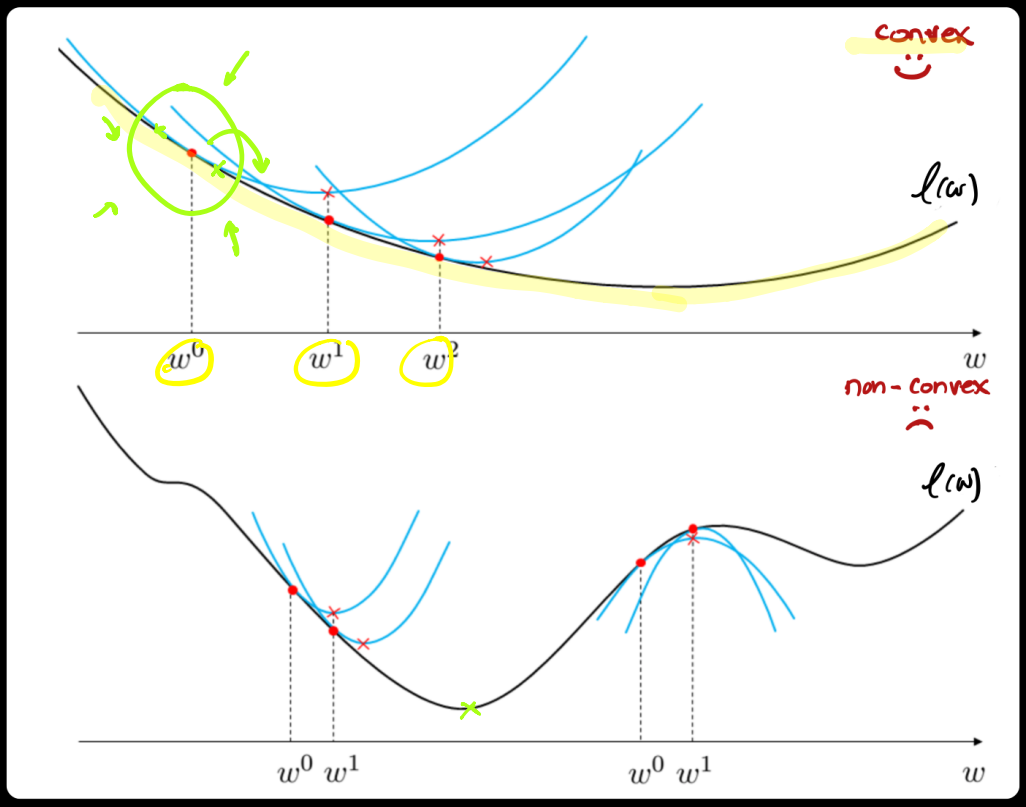
- Newton's method uses second-order (quadratic) approximation to locate the optimum w^* of $\ell(w)$.

Steps:

1. Begin at an initial point w^0
2. Travel to a stationary point (min/max) of the approximating quadratic.
3. Hop back into the function ℓ







* Second-order Taylor approximation of $\ell(w)$ around w^0:

$$h(w) = \ell(w^0) + \nabla\ell(w^0)^T(w - w^0) + \frac{1}{2}(w - w^0)^T\nabla^2\ell(w^0)(w - w^0)$$

(*) Compute approximating quadratic function:

(1) Set gradient to zero "0"

$$\nabla h(w) = \nabla\ell(w^0) + \nabla^2\ell(w^0)(w - w^0) = 0_{N\times 1}$$

$$\Rightarrow \nabla^2\ell(w^0)\,w = \nabla^2\ell(w^0)\,w^0 - \nabla\ell(w^0) \quad (\in \mathbb{R}^N)$$

$$A \quad w = b$$

$A \in \mathbb{R}^{N\times N}$, $b \in \mathbb{R}^{N\times 1}$

$\Rightarrow$ Solve N equations

$$w^1 = \begin{cases} \\ \vdots \end{cases}$$

② find next point w^1

⇒ Sol° is the point w^1 traveled to by Newton's method.

③ repeat steps ① & ② until convergence

Solve for:

$$\nabla^2 \ell(w^1)\, w = \nabla^2 \ell(w^1)\, w^1 - \nabla \ell(w^1)$$

↓

Sol° is w^2

→ at k^{th} step (estimate w^k)

$$\nabla^2 \ell(w^{k-1})\, w^k = \nabla^2 \ell(w^{k-1})\, w^{k-1} - \nabla \ell(w^{k-1})$$

- what happens when $\nabla \ell(w^{k-1}) = 0_{N\times 1}$?

$$w^k = w^{k-1}$$

Algorithm 2.2 Newton's method

Input: twice differentiable function ℓ, and initial point $\mathbf{w}^0$

$k = 1$

Repeat until stopping condition is met:

Solve the system $\nabla^2 \ell\left(\mathbf{w}^{k-1}\right) \mathbf{w}^k = \nabla^2 \ell\left(\mathbf{w}^{k-1}\right) \mathbf{w}^{k-1} - \nabla \ell\left(\mathbf{w}^{k-1}\right)$ for $\mathbf{w}^k$.

$k \leftarrow k + 1$

$[\nabla^2 \ell(w^{k-1})]^{-1} x$

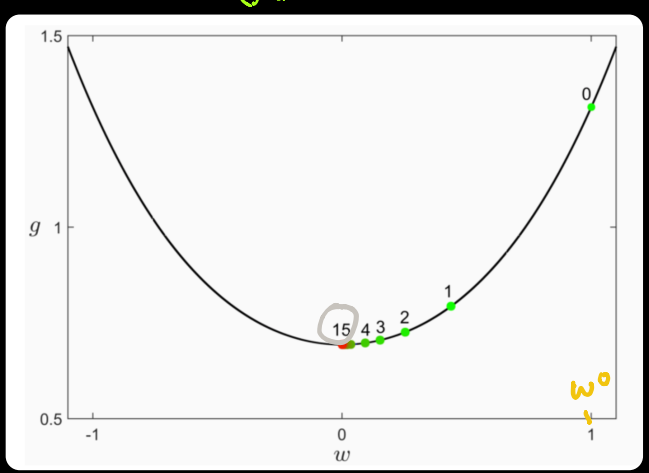
- If $\nabla^2 \ell(w^{k-1})$ is invertible? $\{ [\nabla^2 \ell(w^{k-1})]^{-1}$

$x\, x^{-1} = x \times \frac{1}{x} = 1$

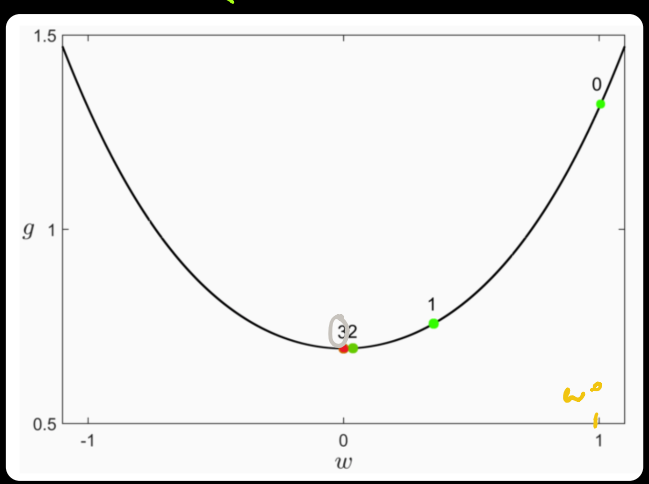
$$w^k = w^{k-1} - [\nabla^2 \ell(w^{k-1})]^{-1} \nabla \ell(w^{k-1})$$

α

gradient descent (first order)



Newton method (second order)



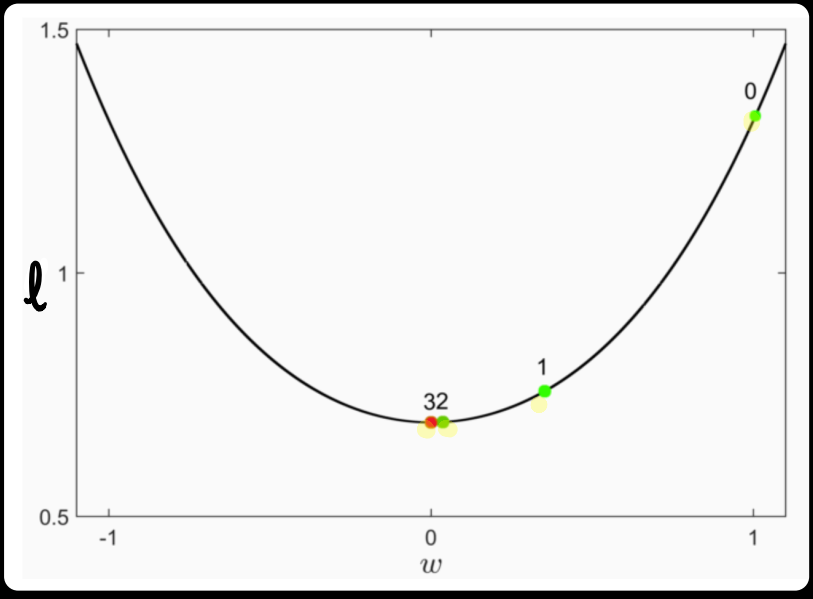
(+) { if ℓ is convex $\Rightarrow$ faster
no α

(-) { non-convex $\Rightarrow$ not converge
inverting the Hessian matrix is computationally expensive

$10^k \times 10^k$

Example 1

$$\begin{cases} w \in \mathbb{R} \text{ (scalar) univariate loss function} \\ \ell(w) = \log(1 + e^{w^2}) \end{cases}$$



- $\ell''(w) = \dfrac{2e^{w^2}(2w^2 + e^{w^2} + 1)}{(1 + e^{w^2})^2}$
- $w^0 = 1$
- $|\ell'(w)| < \varepsilon = 10^{-3} = 0.001$

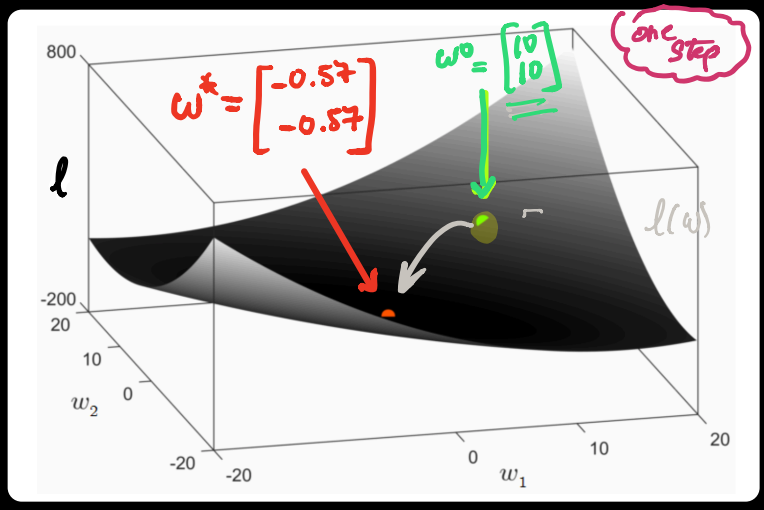
$$w^k = w^{k-1} - \frac{\ell'(w^{k-1})}{\ell''(w^{k-1})} > 0$$

Example 2

$$l(w) = \frac{1}{2} w^T Q w + r^T w + d$$

$Q = \begin{bmatrix} 1 & 0.75 \\ 0.75 & 1 \end{bmatrix}$, $r = [1 \;\; 1]^T$, and $d = 0$

- $w^0 = [10 \;\; 10]^T$
- $\nabla l(w) = Qw + r$
- $\nabla^2 l(w) = Q$ ←



- Write the algebric expression for w^1 & w^2. What do you notice ?

$$\Rightarrow \nabla^2 l(w^{k-1})\, w^k = \nabla^2 l(w^{k-1})\, w^{k-1} - \nabla l(w^{k-1})$$

$$Q \times w^1 = Q \times w^0 - Qw^0 - r$$

$k=1 \Rightarrow$

$$\Rightarrow Q w^1 = -r$$

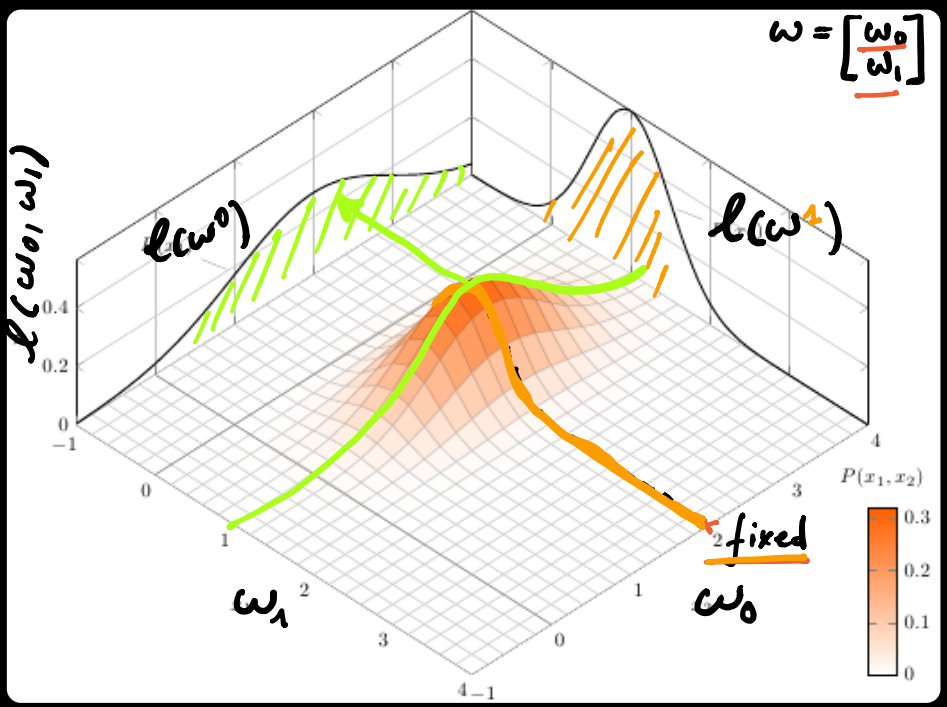
$$\Rightarrow \boxed{w^1 = -Q^{-1} r} \qquad \boxed{w^2 = -Q^{-1} r}$$

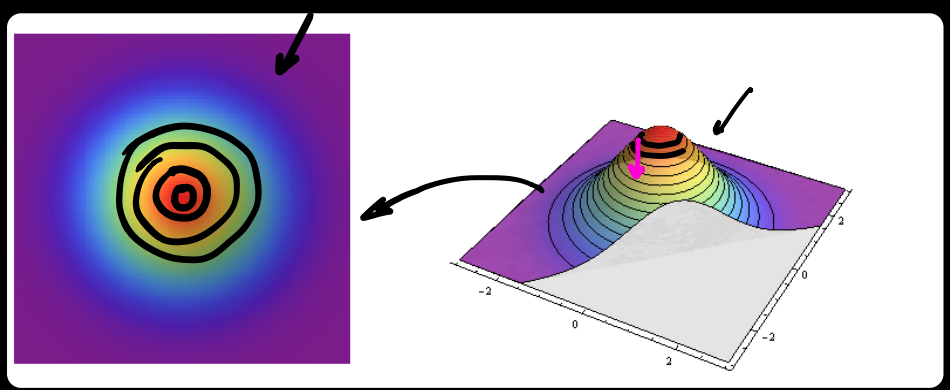
I ♥ Mx [4]

Mathematical notation	Definition
$\mathbf{w}^k = \mathbf{w}^{k-1} - \alpha_k \nabla l(\mathbf{w}^{k-1})$	gradient descent for finding the optimal $\mathbf{w}^*$
$\nabla^2 l(\mathbf{w}^{k-1})\mathbf{w}^k = \nabla^2 l(\mathbf{w}^{k-1})\mathbf{w}^{k-1} - \nabla l(\mathbf{w}^{k-1})$	Newton's method update rule (second order optimization)
$\mathbf{w}^k = \mathbf{w}^{k-1} - [\nabla^2 l(\mathbf{w}^{k-1})]^{-1} \nabla l(\mathbf{w}^{k-1})$	Newton's method update rule when the Hessian matrix $\nabla^2 l(\mathbf{w}^{k-1})$ is invertible

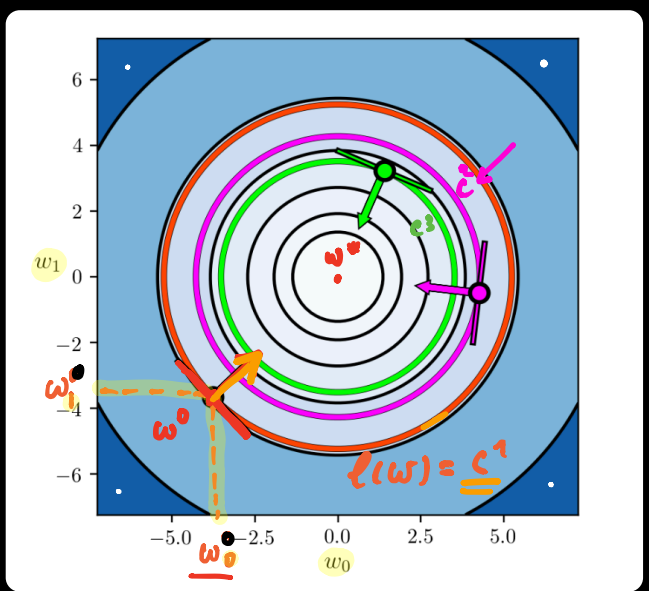
[Matlab / Python demo]

MLR - exercise 2.14





$\ell(w) = w_0^2 + w_1^2 + 2$



$\nabla \ell$ = normal vector to the level curve $\ell(w) = c^1$

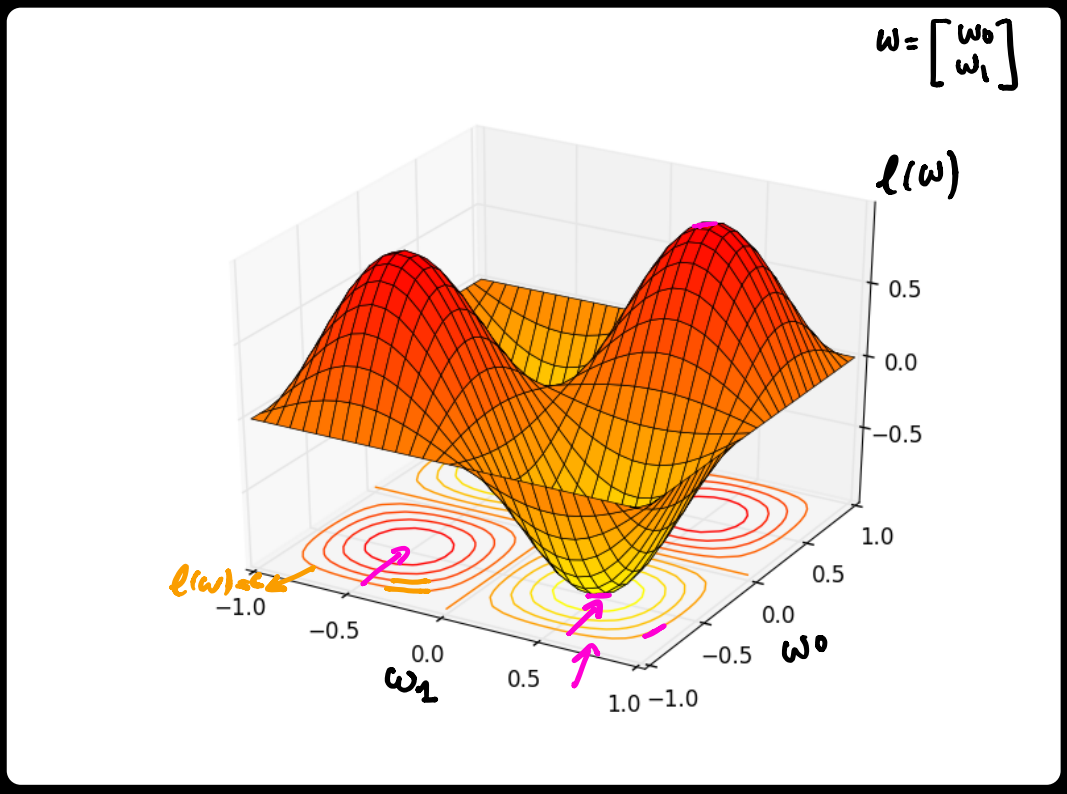
IMPORTANT PROPERTY OF THE NEGATIVE GRADIENT DIRECTION

It is always perpendicular to the contours (level curves) of a function.

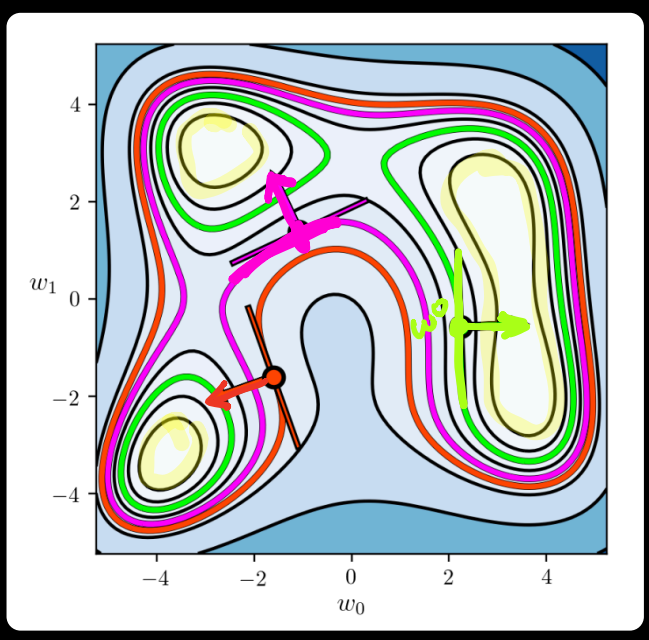
Universally true statement

↓

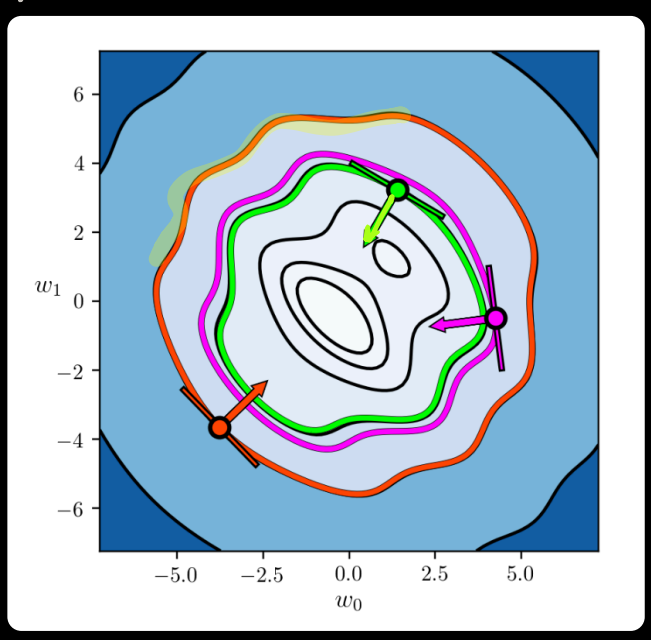
It holds for any function and at all of its inputs.



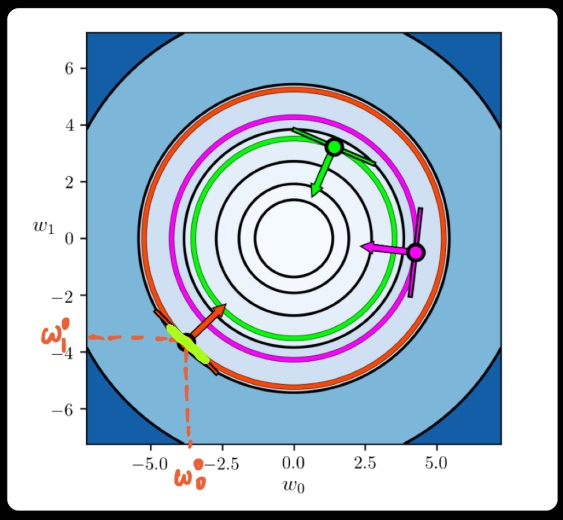
$$\ell(w) = (w_0^2 + w_1 - 11)^2 + (w_0 + w_1^2 - 6)^2$$



$$\ell(w) = w_0^2 + w_1^2 + 2\sin(1.5(w_0 + w_1)^2) + 2$$



$$\ell(w) = w_0^2 + w_1^2 + 2$$



Let us confirm this fundamental property using rigorous mathematical reasoning!

① $\ell(w)$ is differentiable at some input point $a \Rightarrow$ a lies on the contour points where $\ell(w) = \ell(a) = c$.

② Let us take another point b close to a then essentially since:

$$\nabla \ell(a)^T (b - a) = 0$$

$\nabla \ell(a)$: gradient is perpendicular to the

$(b - a)$: vector

Since the 1st order approximation $h(b)$ around a:

$$h(b) = \ell(a) + \nabla \ell(a)^T (b - a)$$

$h(b)$ ← same → $\ell(a)$

$\nabla \ell(a)^T (b - a)$: 0

$$\ell'(b) = \lim_{b \to a} \frac{\ell(b) - \ell(a)}{b - a}$$

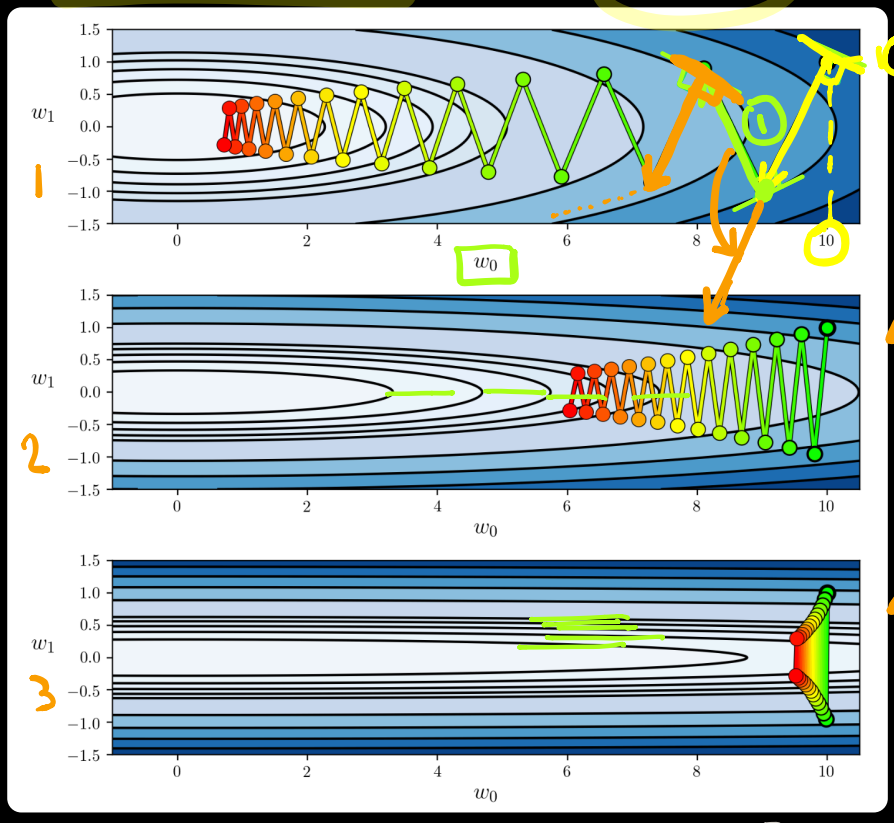
② The 'zig-zagging' behavior of gradient descent

☹ It slows down convergence due to the oscillation of the negative gradient direction

Example:

$$\begin{cases} \ell(w) = a + b^T w + w^T C w \; ; \; w \in \mathbb{R}^2 \\ a = 0 \; , \; b = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \end{cases}$$

$f(w) = w^T C w$, $\alpha = 0.1$, $w^0 = \begin{bmatrix} 10 \\ 1 \end{bmatrix}$, 25 runs.

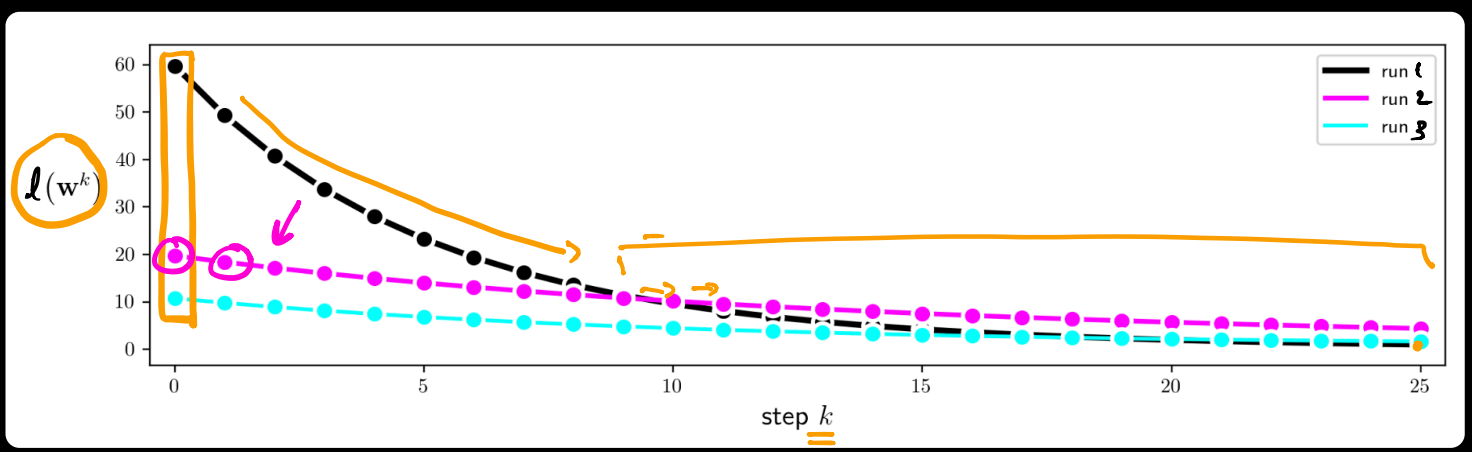


$C = \begin{matrix} w^0 \\ w^1 \end{matrix} \begin{bmatrix} 0.5 & 0 \\ 0 & 12 \end{bmatrix}$ $\quad C = \begin{bmatrix} 0.1 & 0 \\ 0 & 12 \end{bmatrix}$ $\quad C = \begin{bmatrix} 0.01 & 0 \\ 0 & 12 \end{bmatrix}$

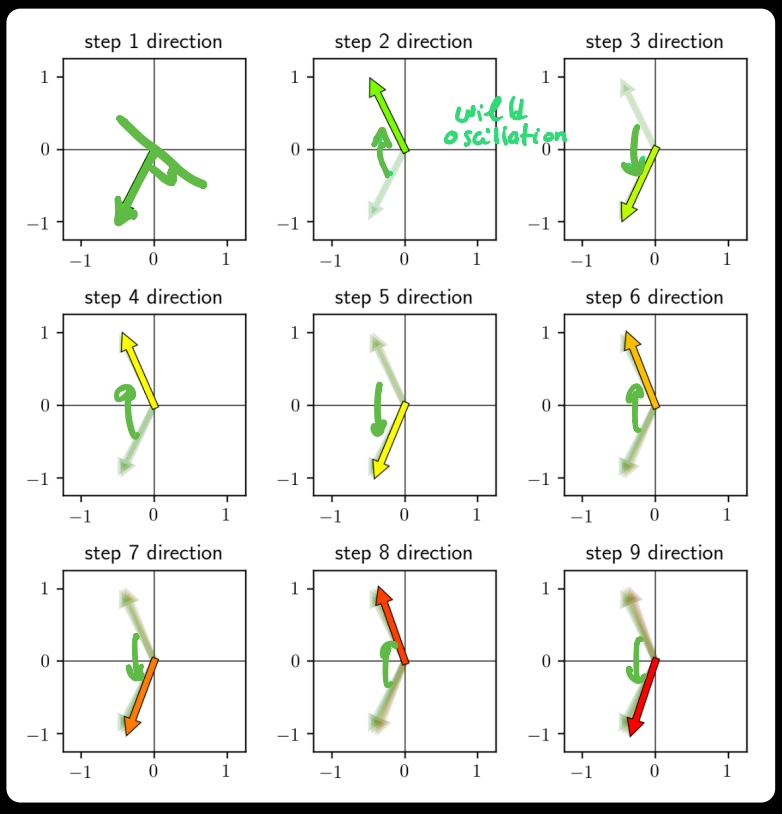
variance along w_0

- Shared global minimum at $w^* = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$
- Zigzagging is caused by the negative gradient direction constantly pointing perpendicular to the contours of the loss function.

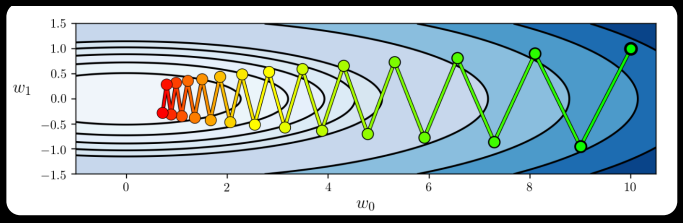
Slow convergence caused by zigzagging



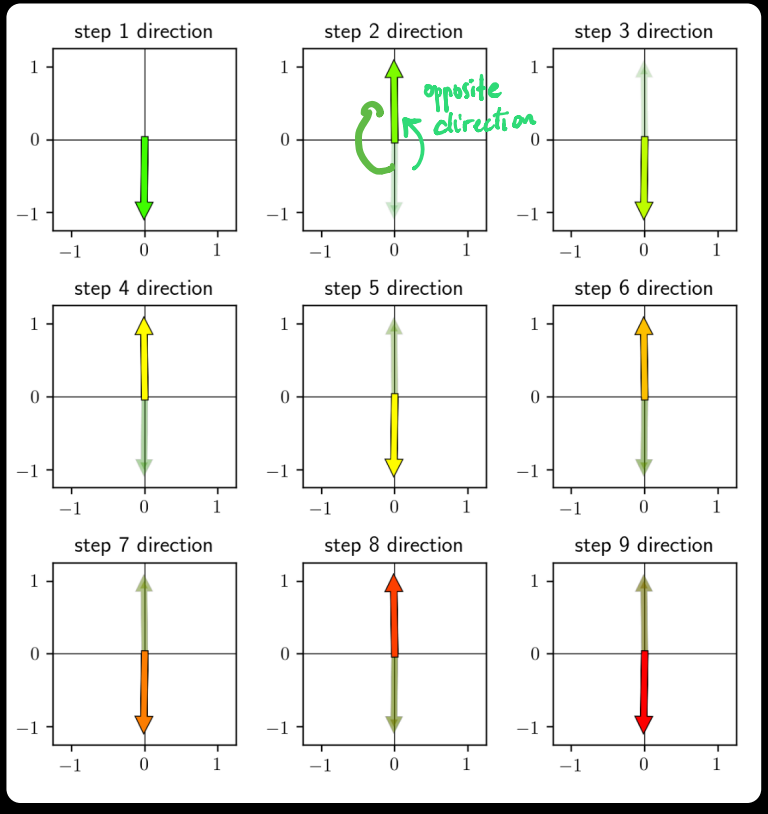
* Plot the descent direction from the first 9 steps.



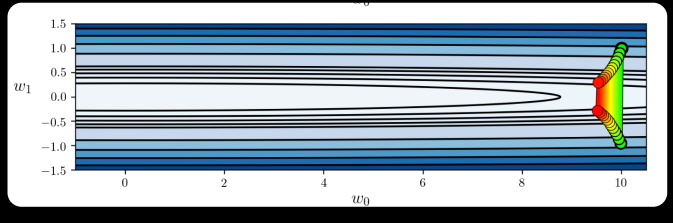
$$\ell(w) = w^T \begin{bmatrix} 0.5 & 0 \\ 0 & 12 \end{bmatrix} w$$



Massive zigzagging in the descent steps.

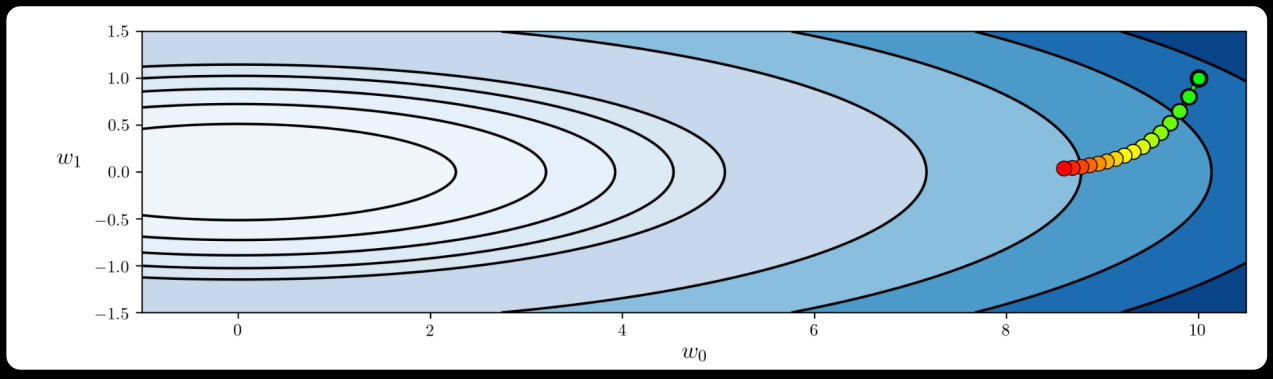


$$\ell(w) = w^T \begin{bmatrix} 0.01 & 0 \\ 0 & 12 \end{bmatrix} w$$

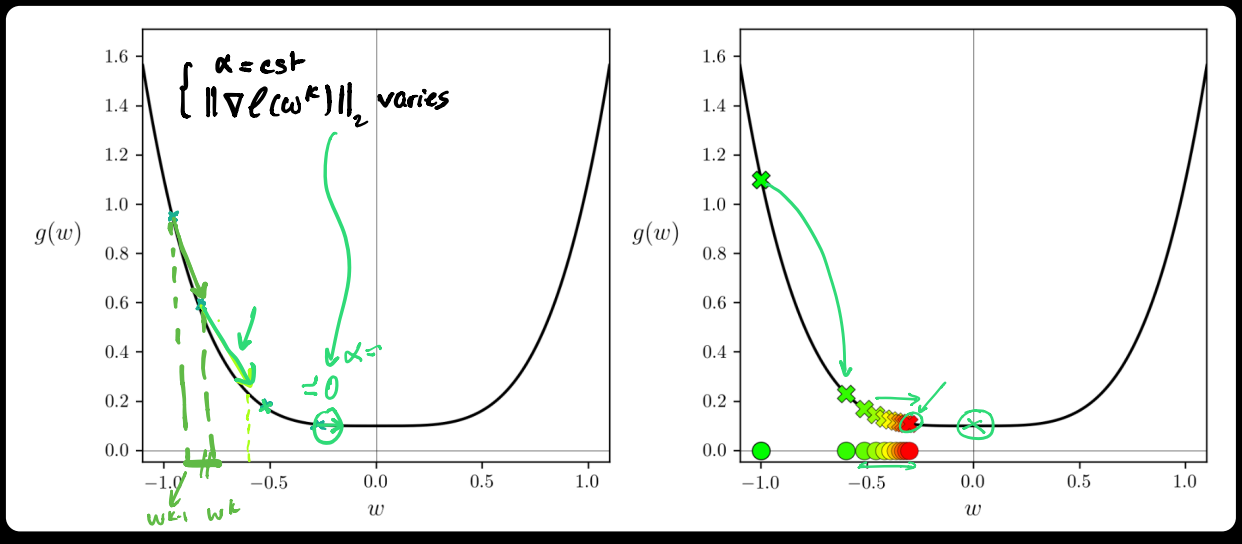


(+) Decreasing steplength value (α) can tone down the zigzagging behavior.

(−) Slow convergence



Issue 2 : the vanishing gradient "vanishing magnitude"



$$w^k = w^{k-1} - \alpha \nabla \ell(w^{k-1})$$

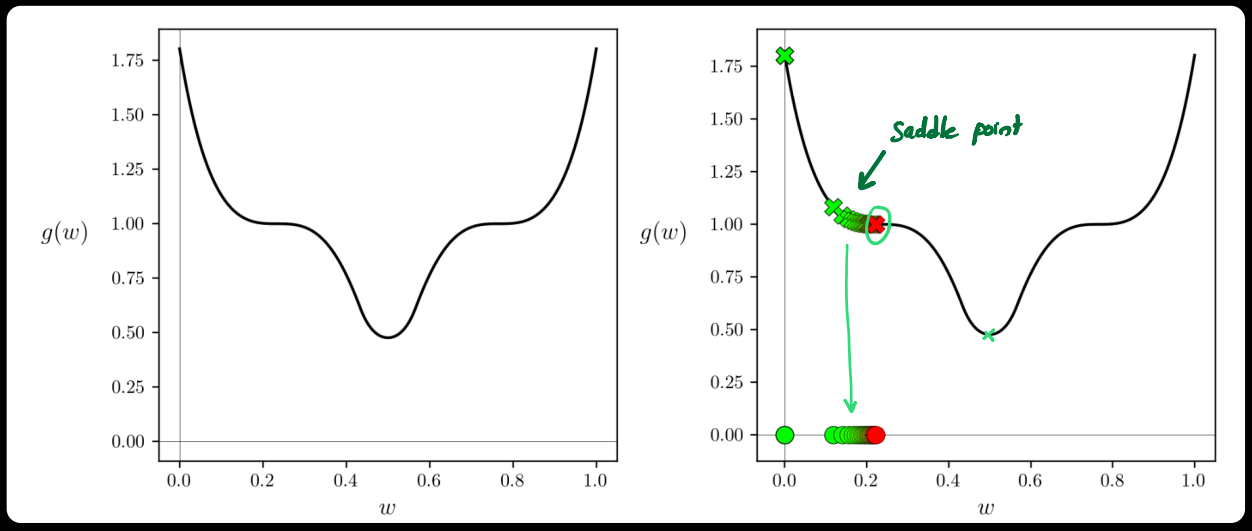
$$\Rightarrow \quad \|w^{k-1} - w^k\|_2 = \alpha \|\nabla \ell(w^{k-1})\|$$

α → 0

magnitude of the gradient

The magnitude of the gradient vanishes at stationary points.

→ Slow-crawling behavior of gradient descent.



☹ Initialization lies on a long narrow valley.
No progress following 1000 steps of gradient descent!