INF721

2024/2

UFV

Deep Learning

L3: Linear Regression

Logistics

Announcements

- I've included lecture notes and readings on the course webpage

Last Lecture

- Machine Learning
  - Supervised Learning
  - Unsupervised Learning
  - Reinforcement Learning
- Supervised Learning Algorithms
  - Hypothesis space
  - Loss function

Lecture outline

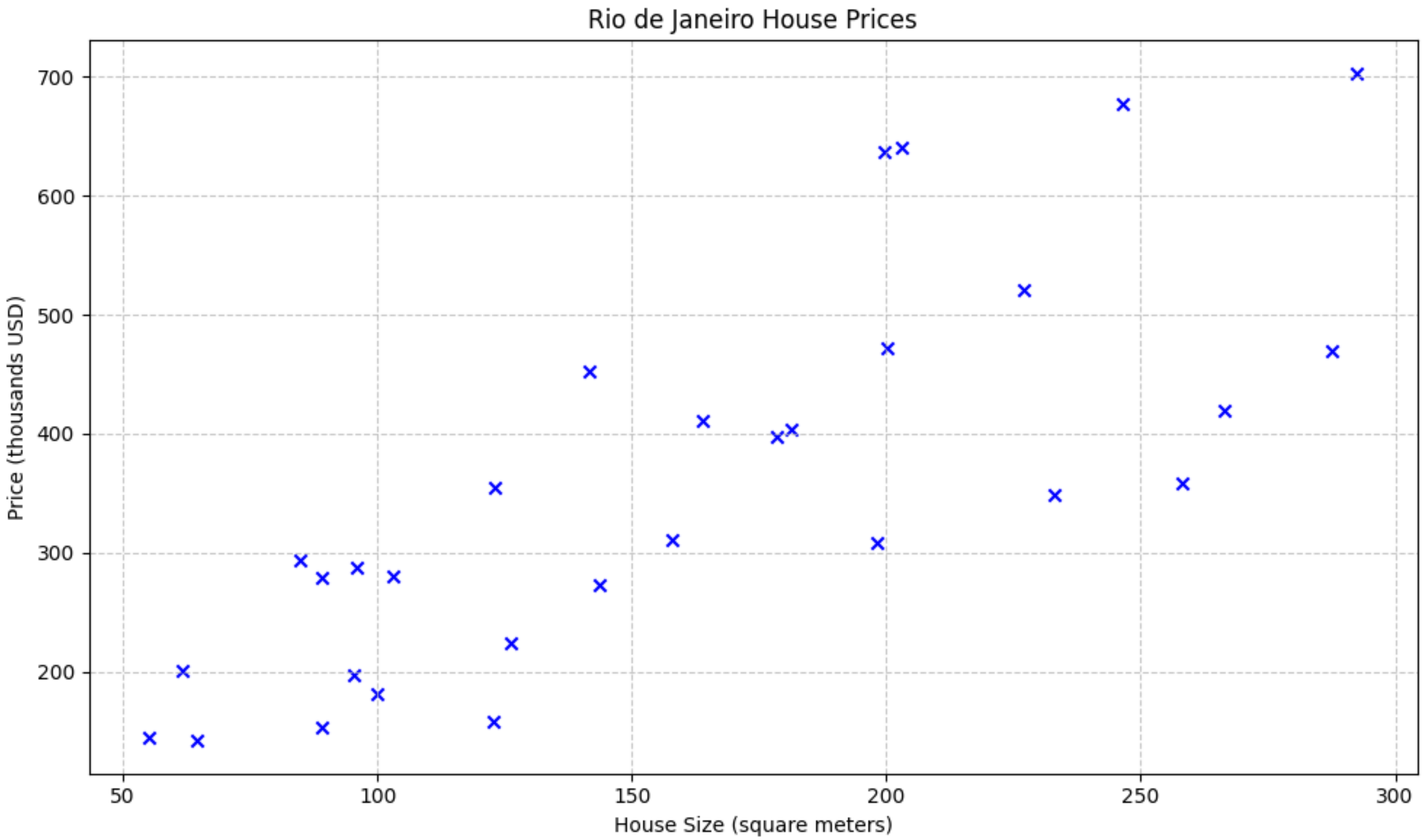
- Univariate Linear Regression
  - Hypothesis space
  - Loss function
- Gradient Descent
  - Derivatives
  - Partial Derivatives
  - Chain Rule
- Gradient Descent for Univariate Linear Regression

Problem 1: House price Prediction

Consider the problem of predicting the price of a house based on its size in squared meters:

Dataset D

x (size m)	y (Price in 1000's USD)
55	144
61	200
84	293
95	196
...	...



Linear Regression

In Linear Regression, we want to find a linear function $h(x)$ that best fits the dataset D

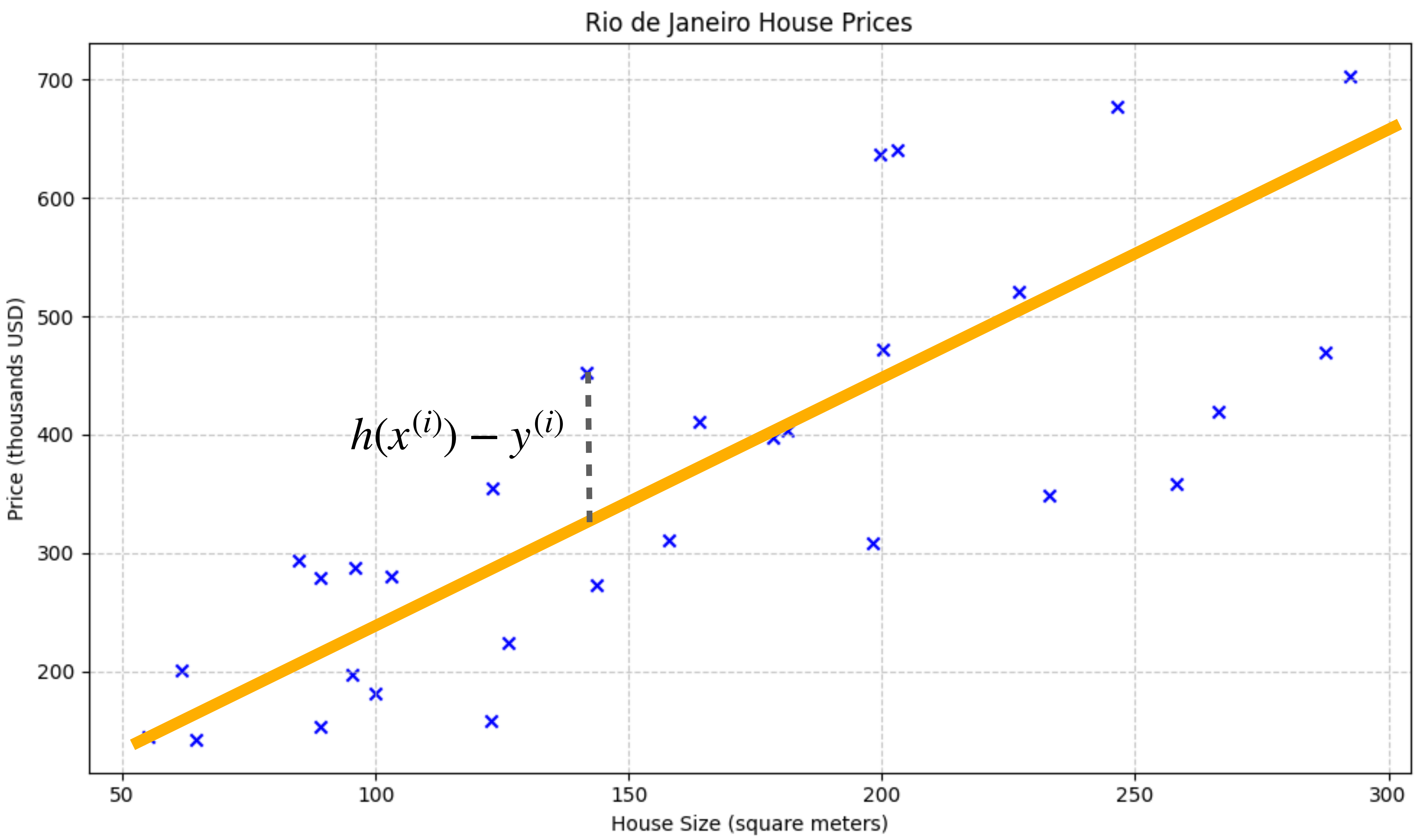
- Hypothesis space H:

  $$h(x) = wx + b$$

- Loss function $L(h)$:

  $$L(h) = \frac{1}{2m} \sum_{i=1}^{n} (h(x^{(i)}) - y^{(i)})^2$$

  Mean Squared Error

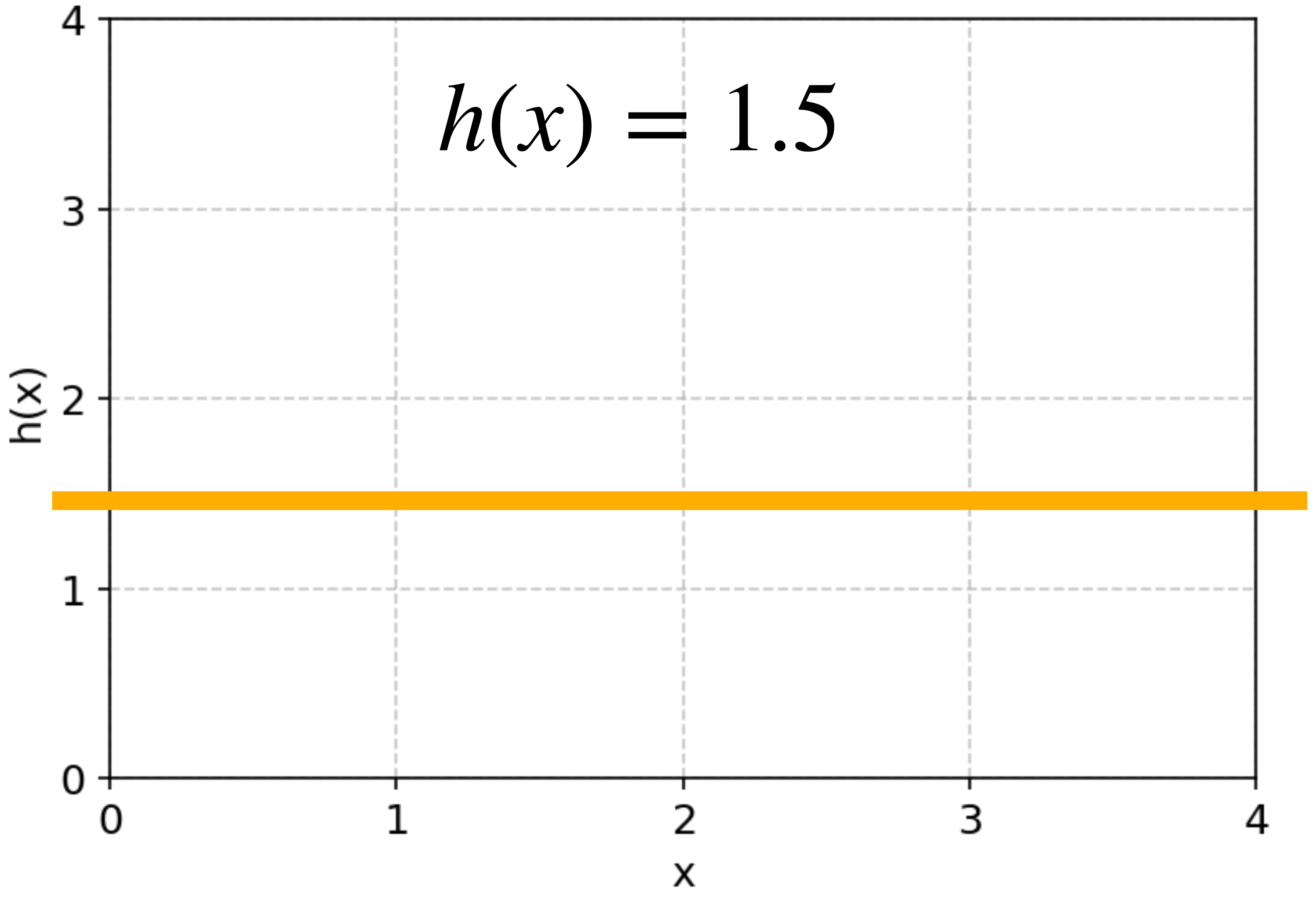


UFV

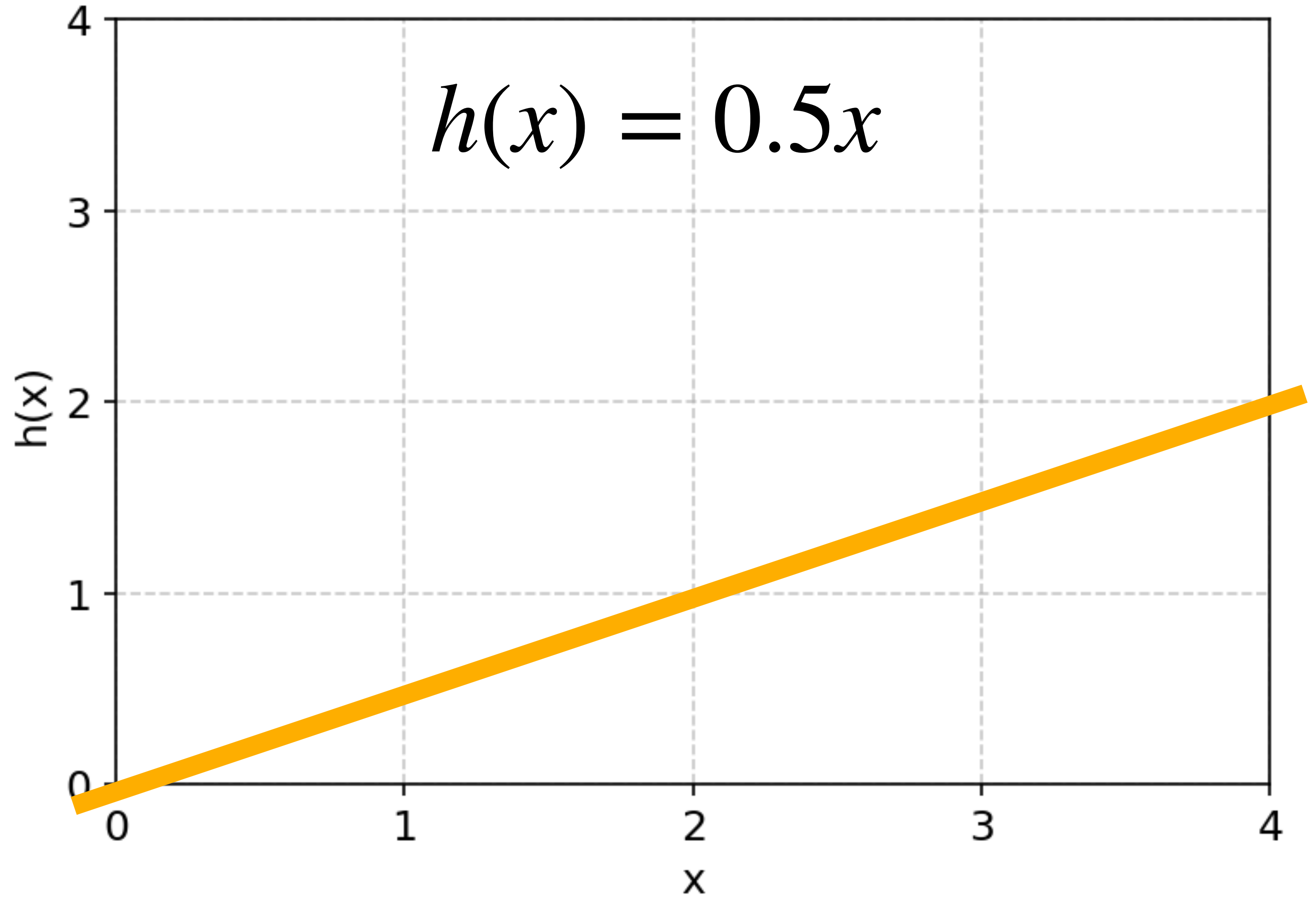
Hypothesis Space

- Hypothesis space H:

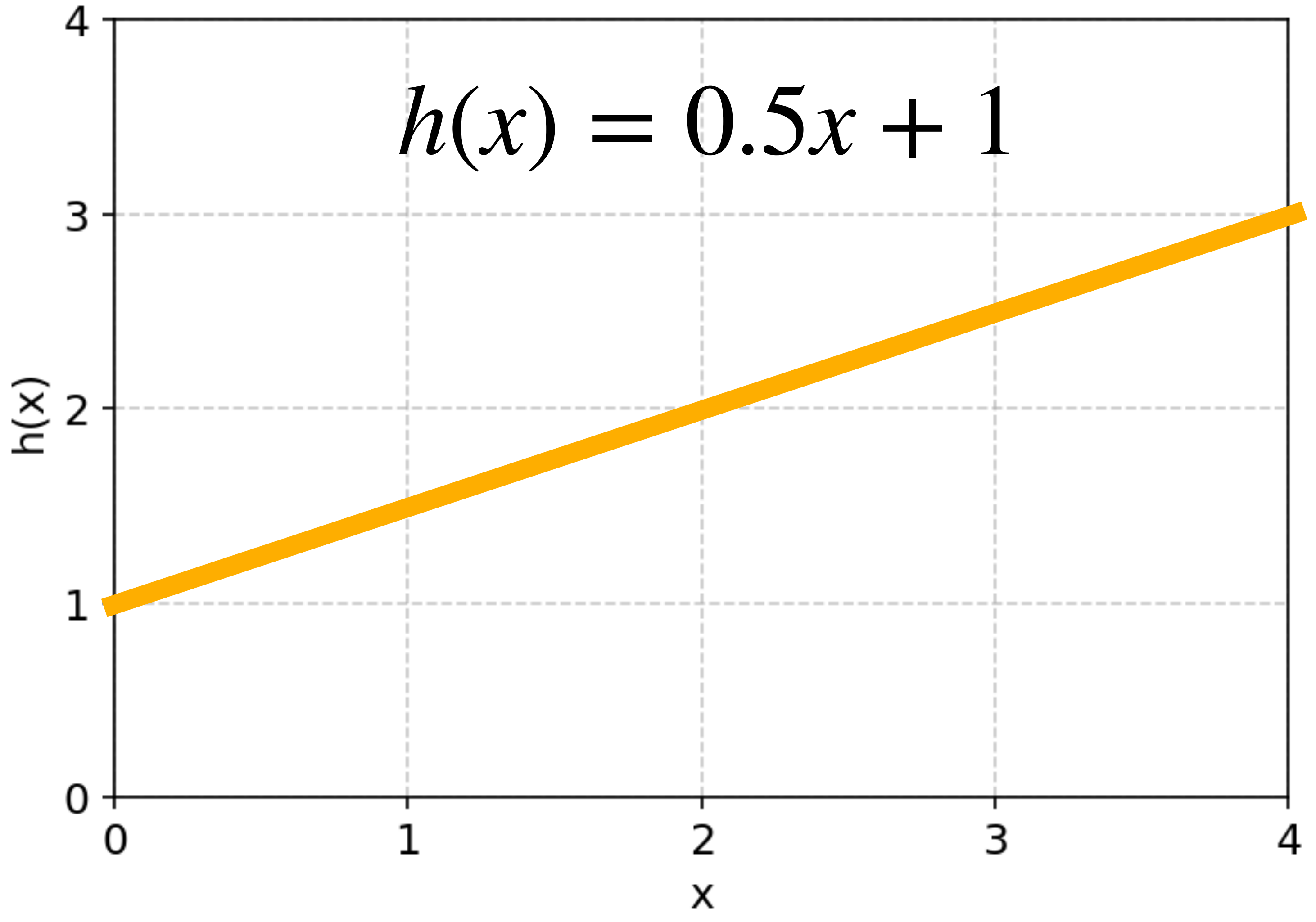
$$h(x) = wx + b$$



$w = 0$
$b = 1.5$



$w = 0.5$
$b = 0$

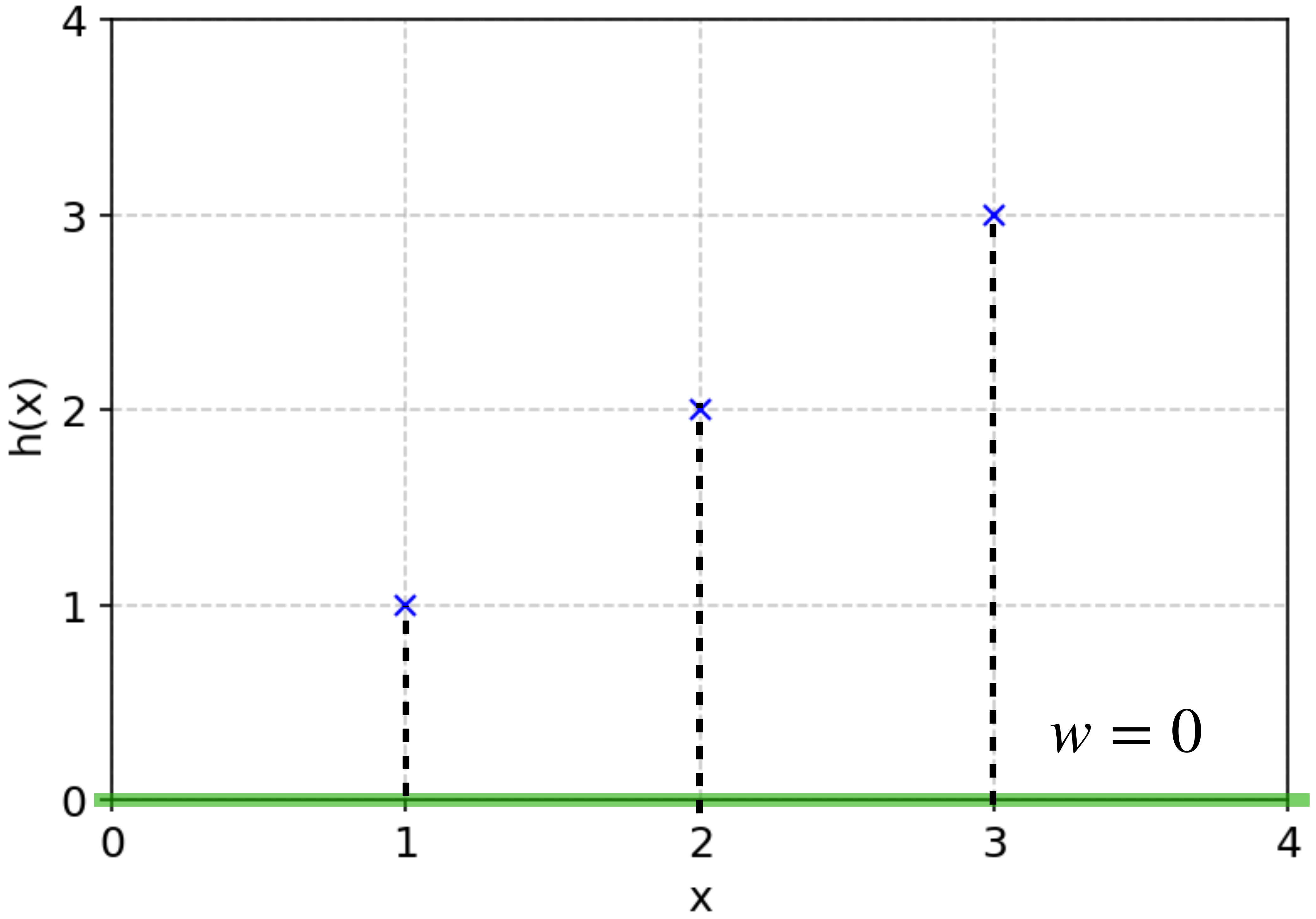


$w = 0.5$
$b = 1$

Loss Function

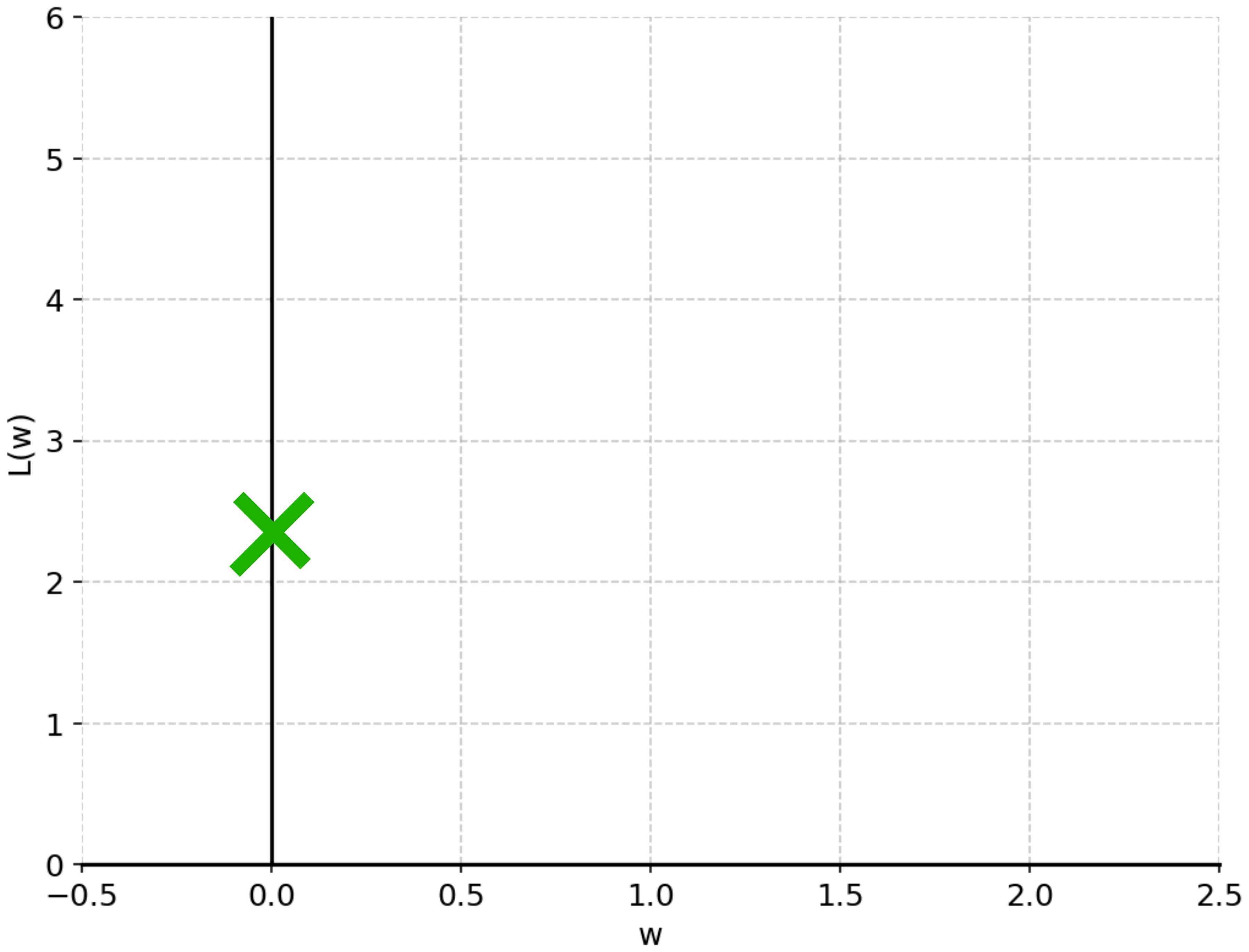
- Simplified hypothesis $(b = 0)$

$$h_w(x) = wx$$



- Mean Squared Error

$$L(h_w) = \frac{1}{2m}\sum_{i=1}^{n}(wx^{(i)} - y^{(i)})^2$$



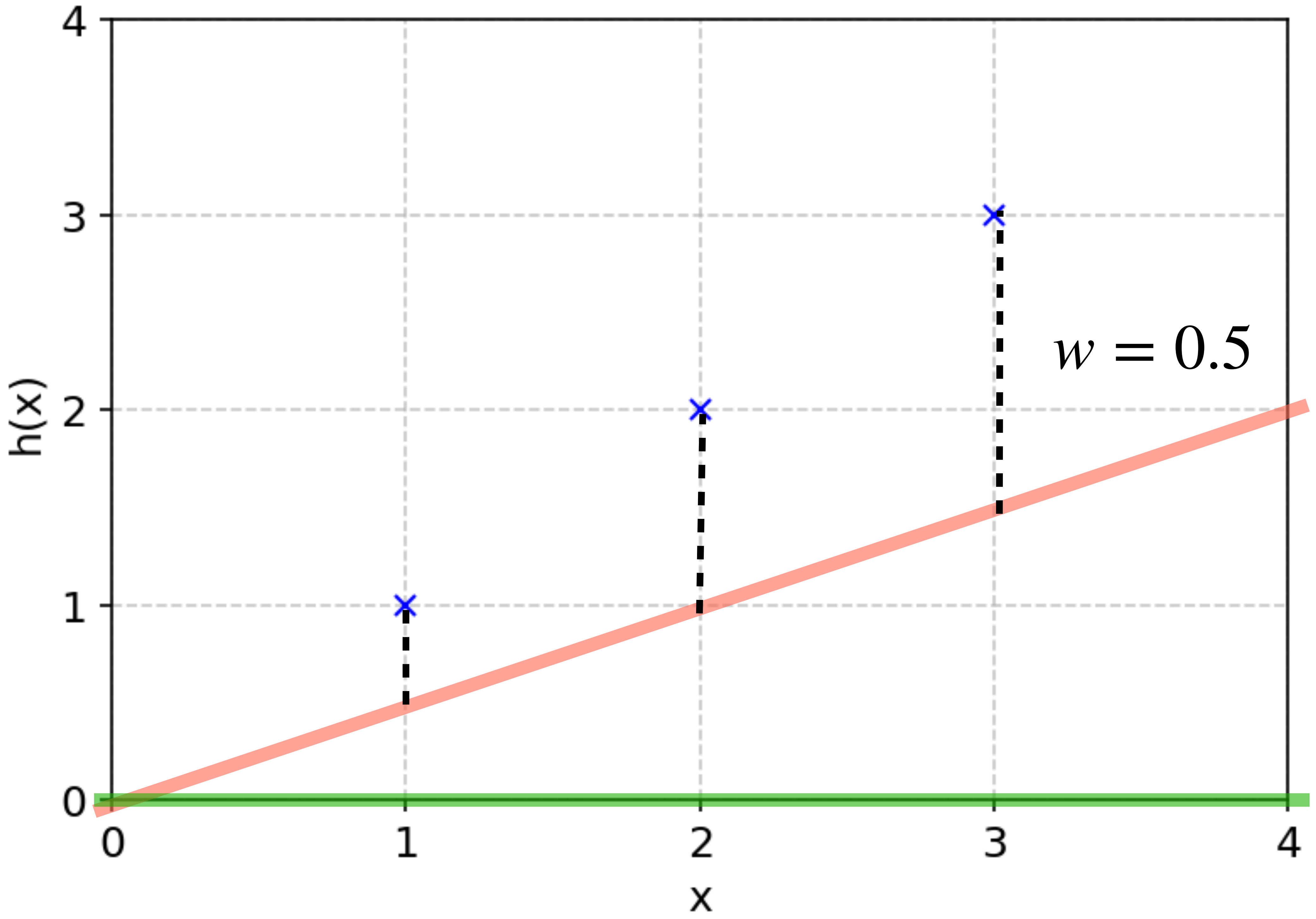
$$L(h_w) = \frac{1}{2 \cdot 3}(0-1)^2 + (0-2)^2 + (0-3)^2 = 2.333$$

UFV

Loss Function

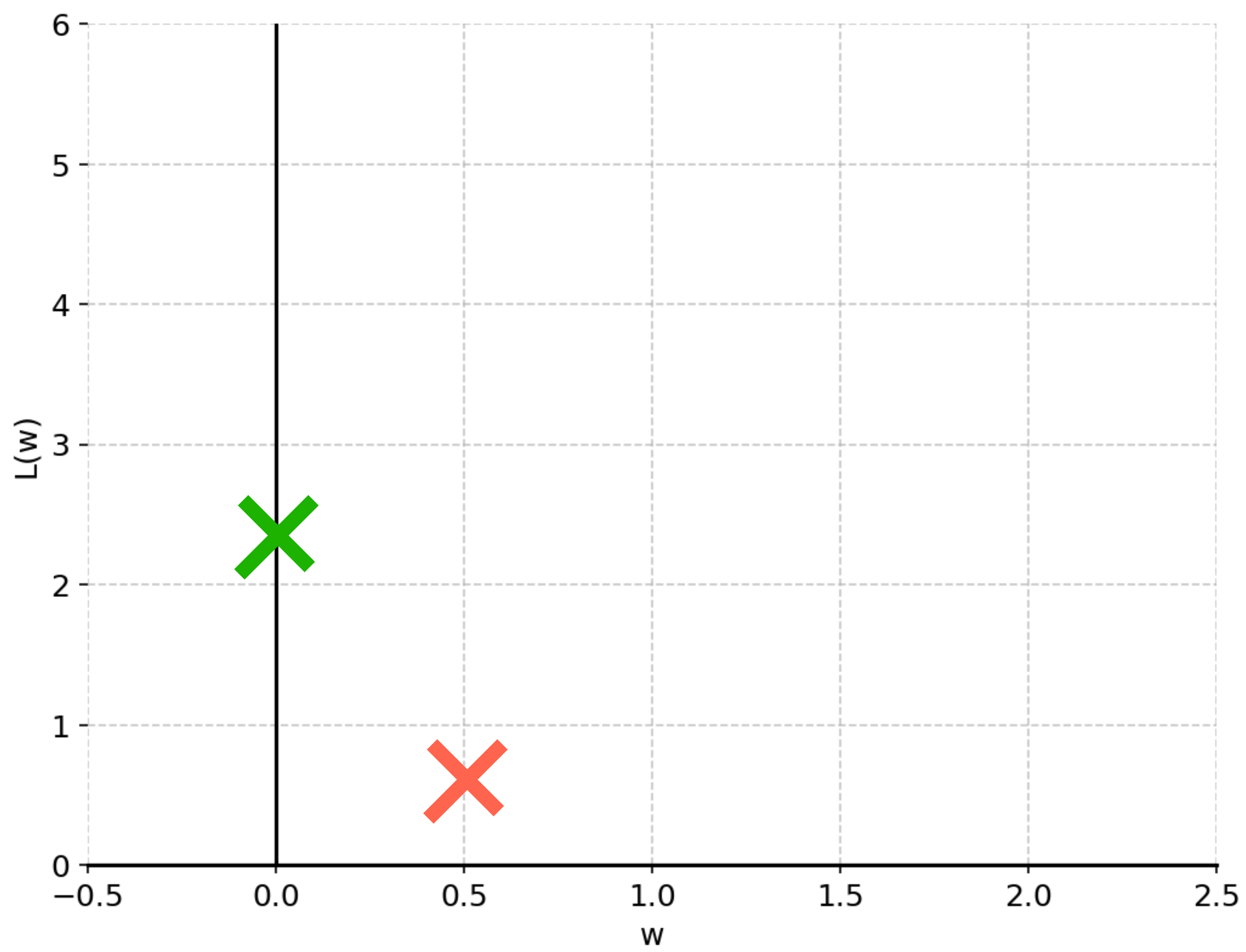
- Simplified hypothesis $(b = 0)$

$$h_w(x) = wx$$



- Mean Squared Error

$$L(h_w) = \frac{1}{2m}\sum_{i=1}^{n}(wx^{(i)} - y^{(i)})^2$$



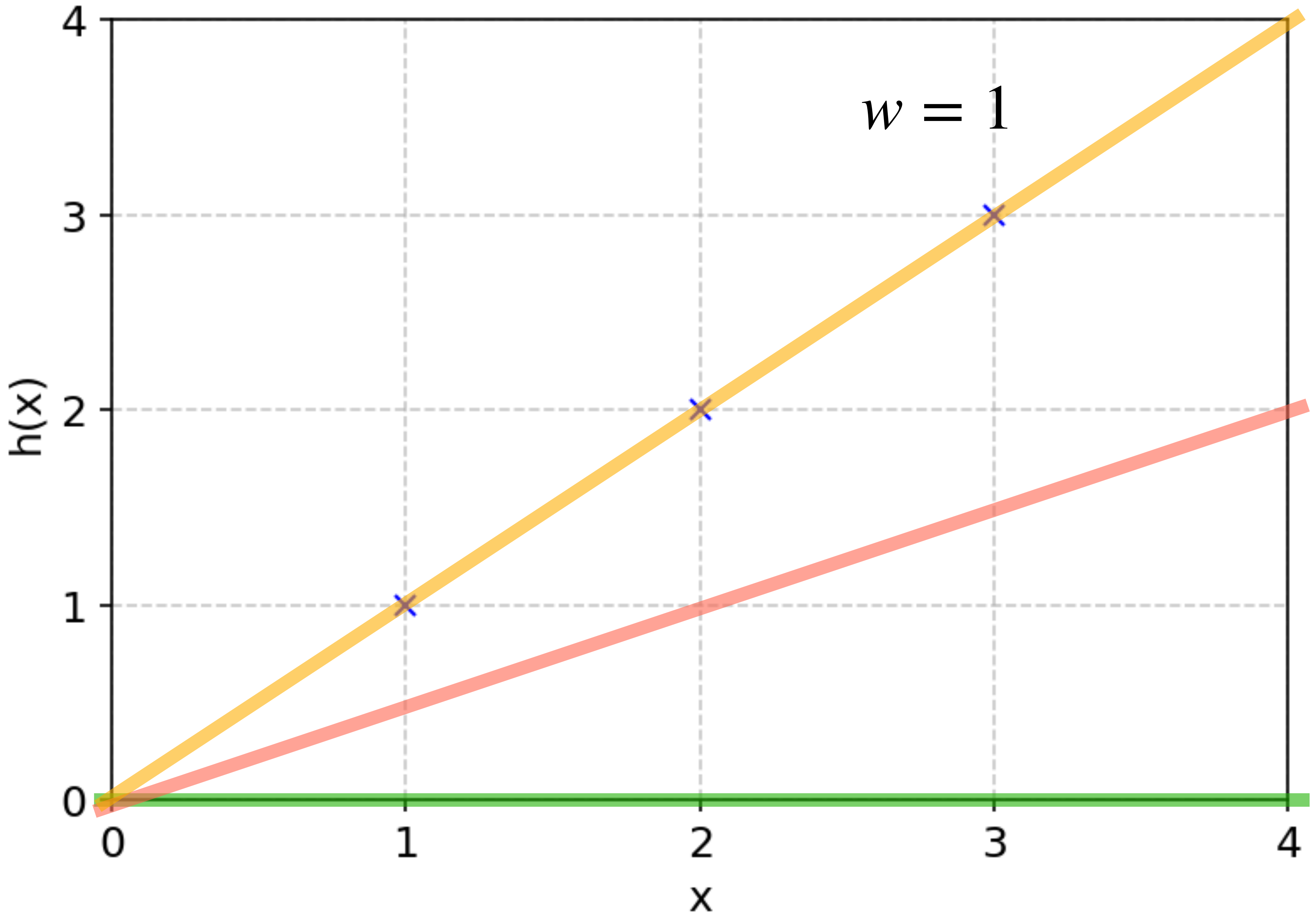
$$L(h_w) = \frac{1}{2 \cdot 3}(0.5 - 1)^2 + (1 - 2)^2 + (1.5 - 3)^2 = 0.583$$

UFV

Loss Function

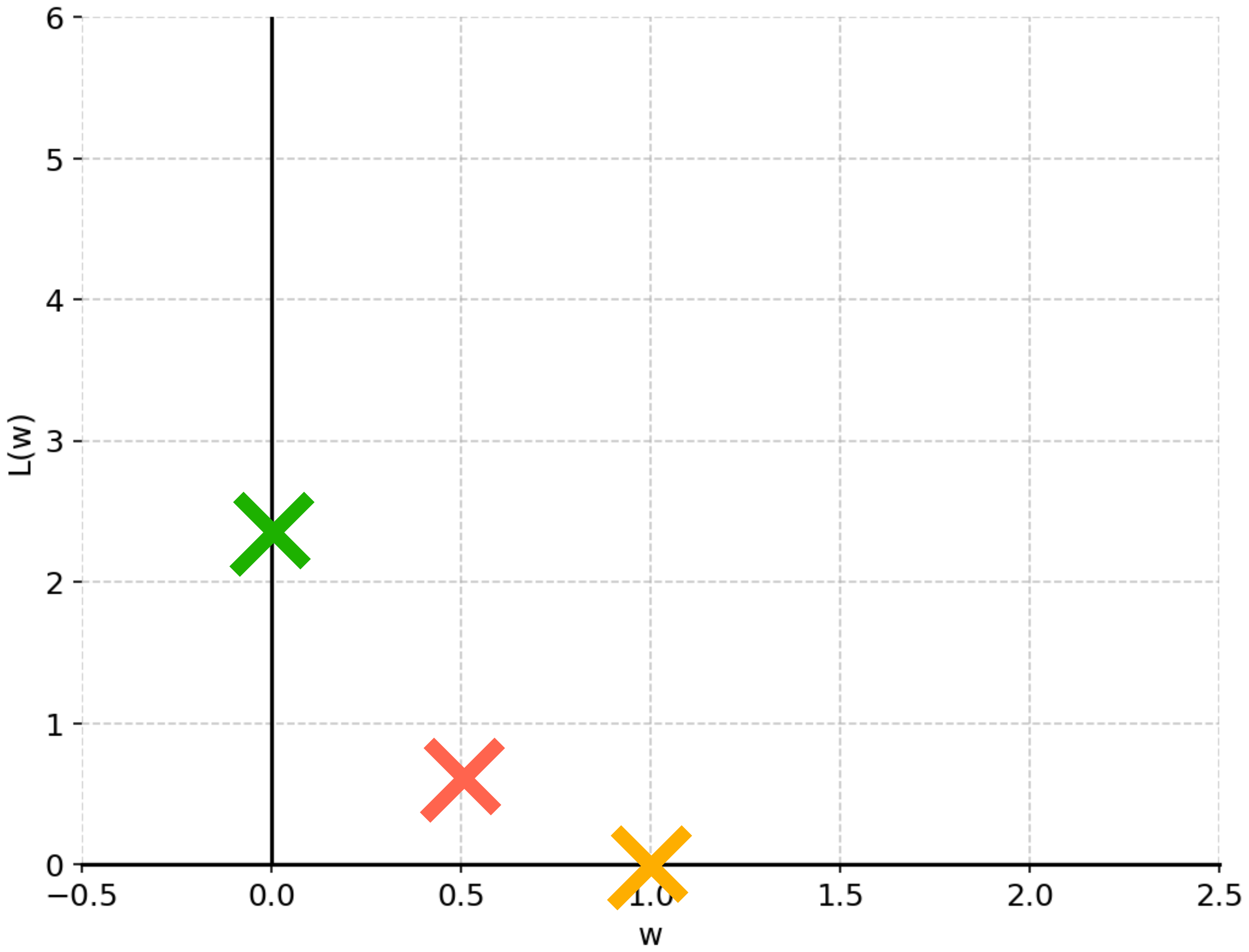
- Simplified hypothesis $(b = 0)$

$$h_w(x) = wx$$



- Mean Squared Error

$$L(h_w) = \frac{1}{2m} \sum_{i=1}^{n} (wx^{(i)} - y^{(i)})^2$$



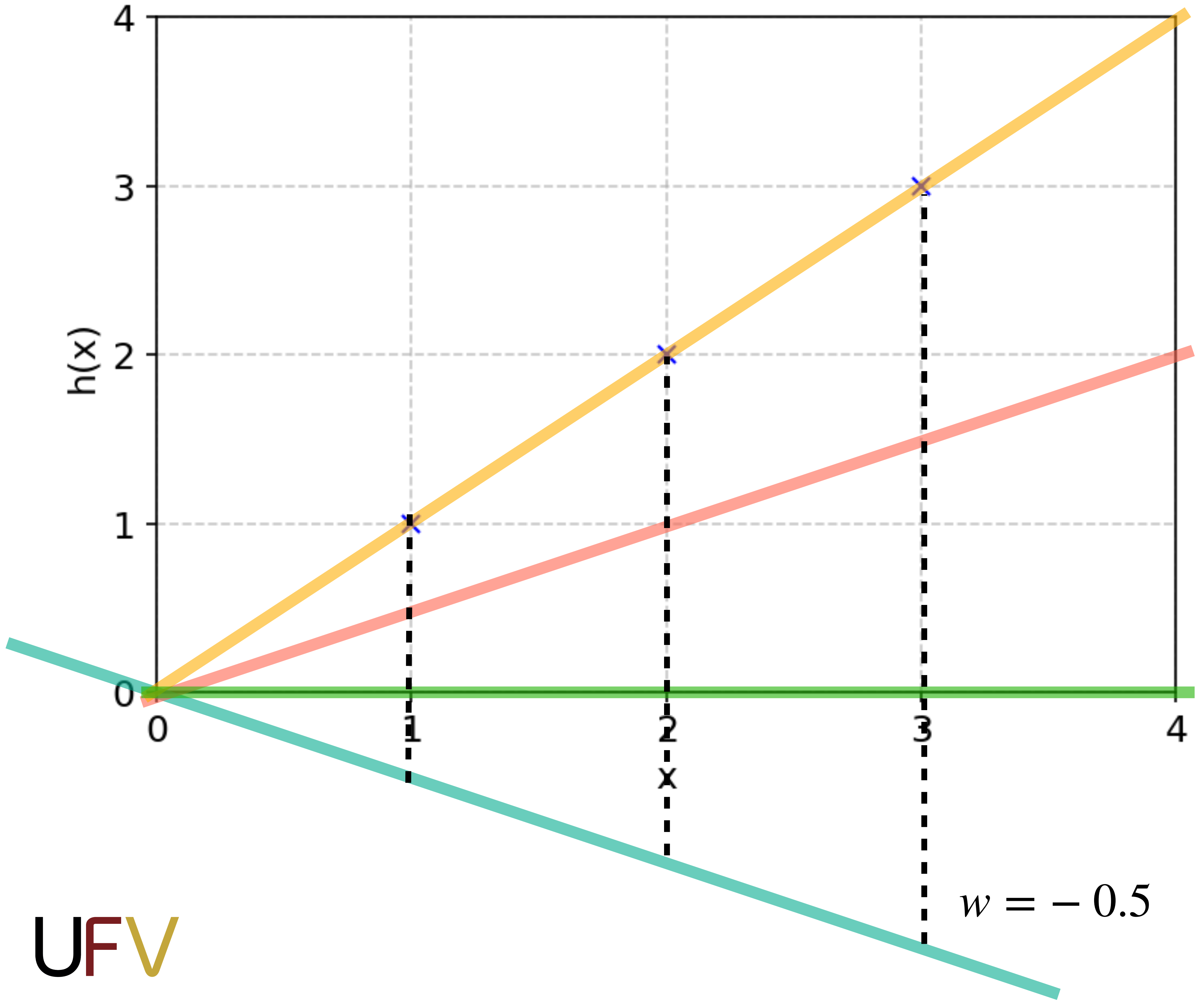
$$L(h_w) = \frac{1}{2 \cdot 3}(1-1)^2 + (2-2)^2 + (3-3)^2 = 0$$

UFV

Loss Function

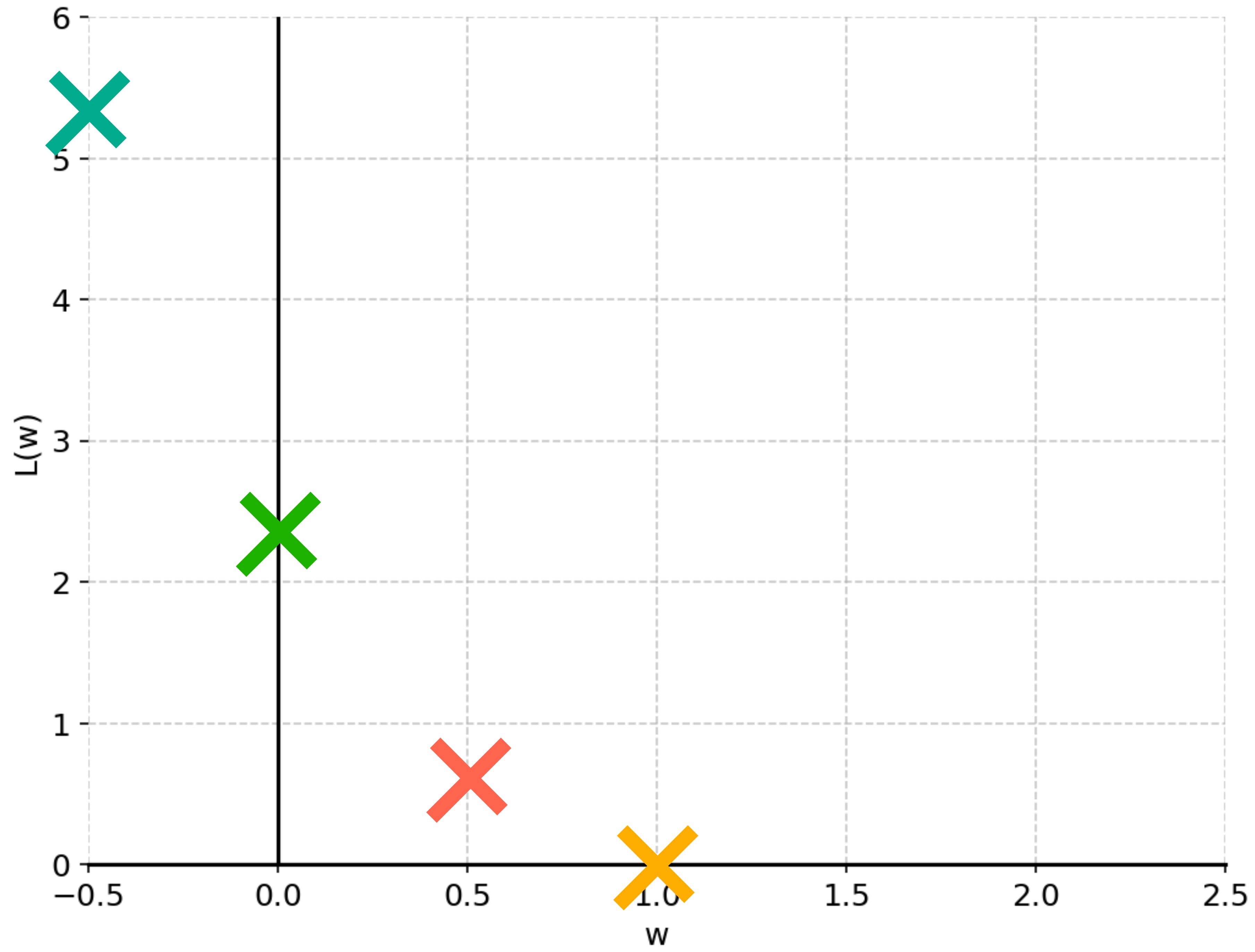
- Simplified hypothesis $(b = 0)$

$$h_w(x) = wx$$



- Mean Squared Error

$$L(h_w) = \frac{1}{2m} \sum_{i=1}^{n} (wx^{(i)} - y^{(i)})^2$$



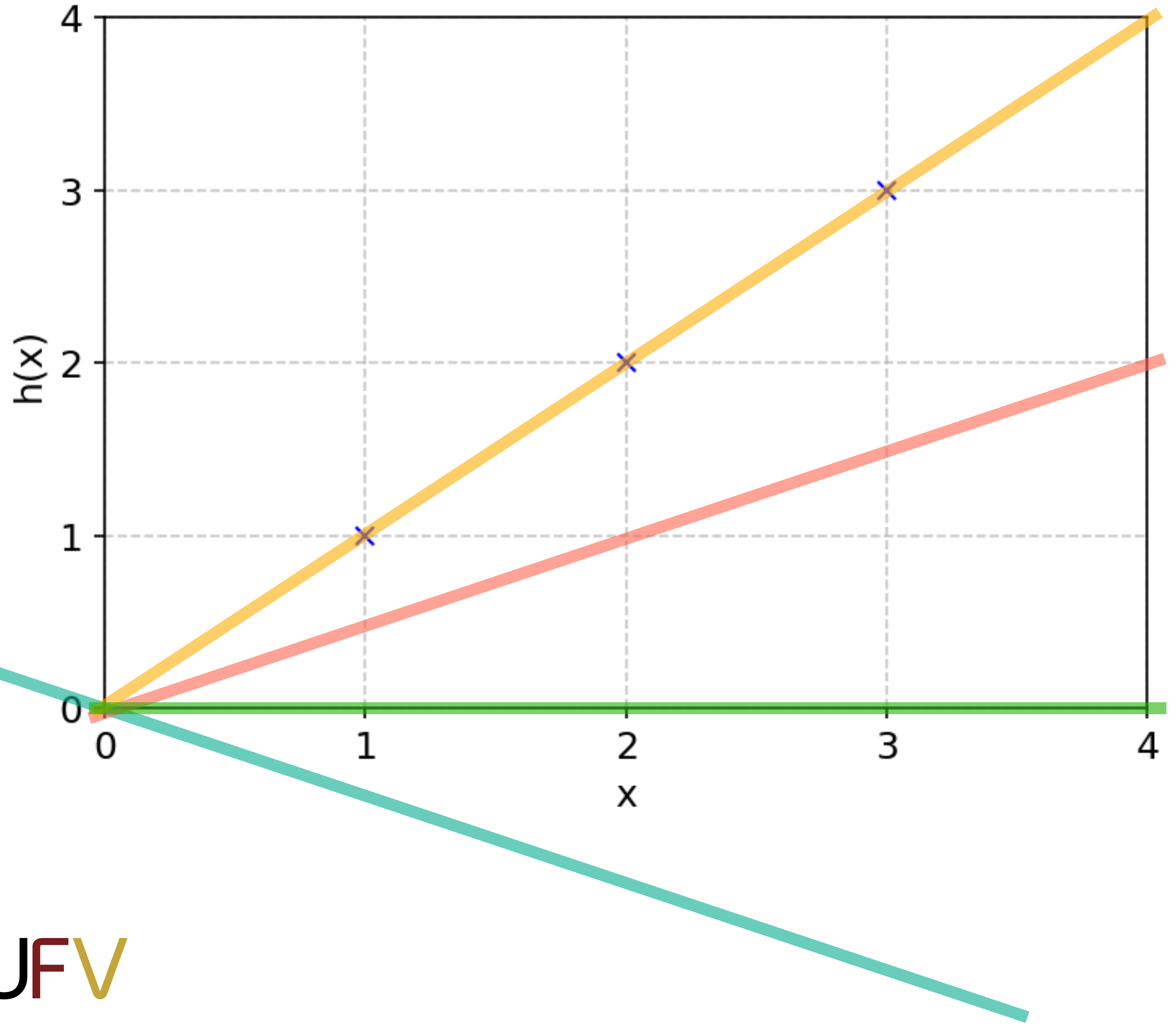
$$L(h_w) = \frac{1}{2 \cdot 3}(-0.5 - 1)^2 + (-1 - 2)^2 + (-1.5 - 3)^2 = 5.25$$

UFV

Loss Function

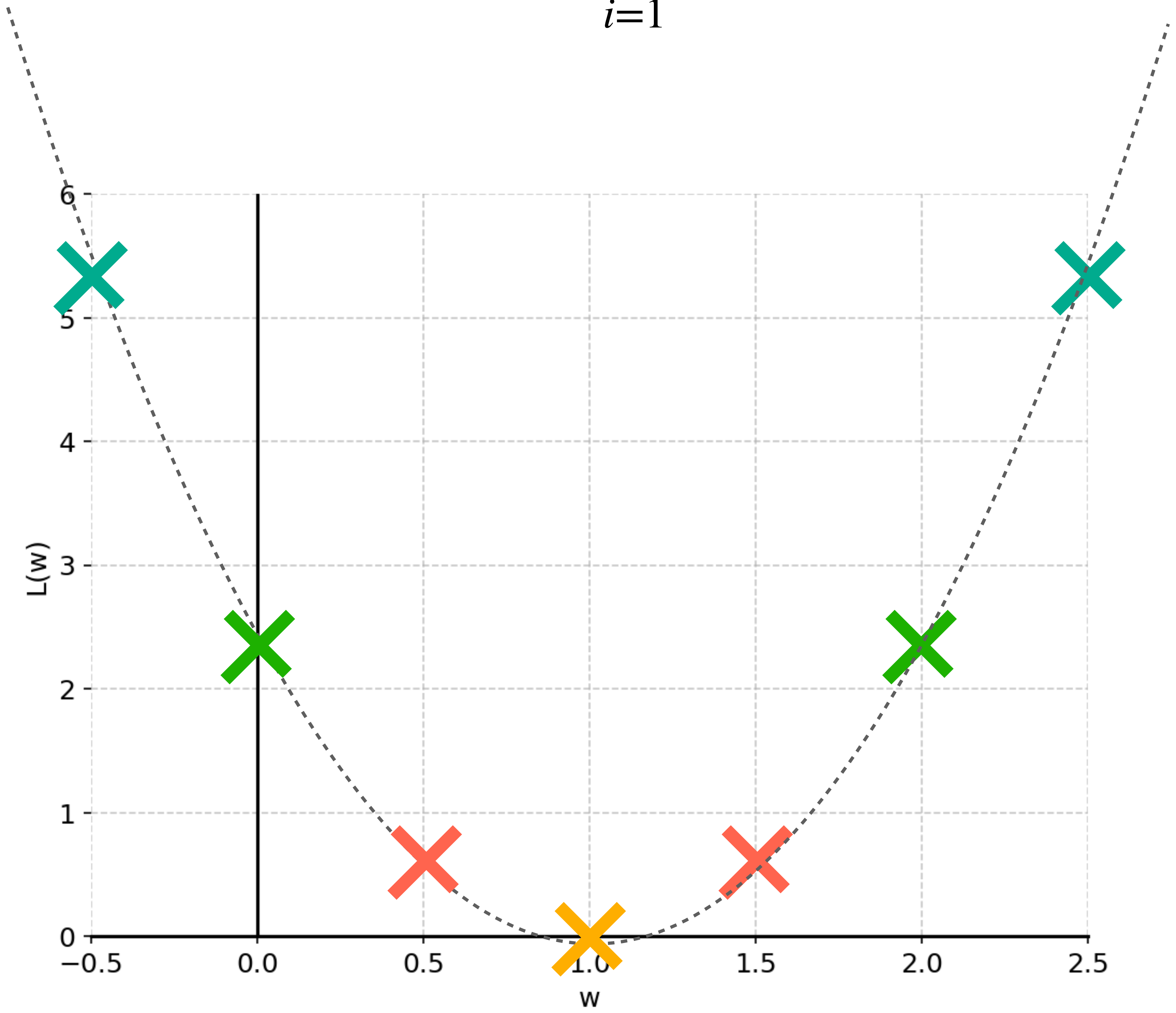
- Simplified hypothesis ($b = 0$)

$$h_w(x) = wx$$



- Mean Squared Error

$$L(h_w) = \frac{1}{2m}\sum_{i=1}^{n}(wx^{(i)} - y^{(i)})^2$$

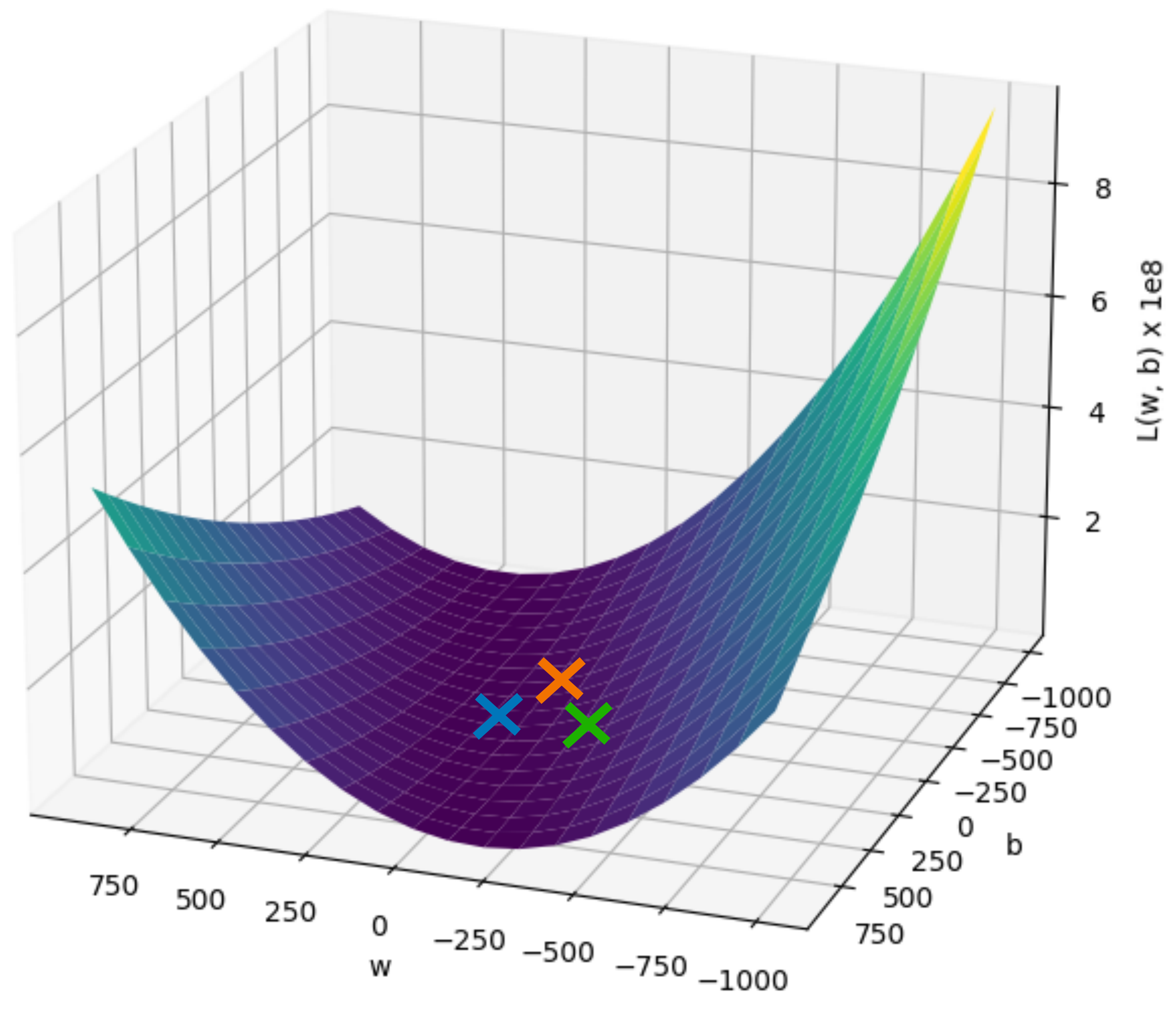


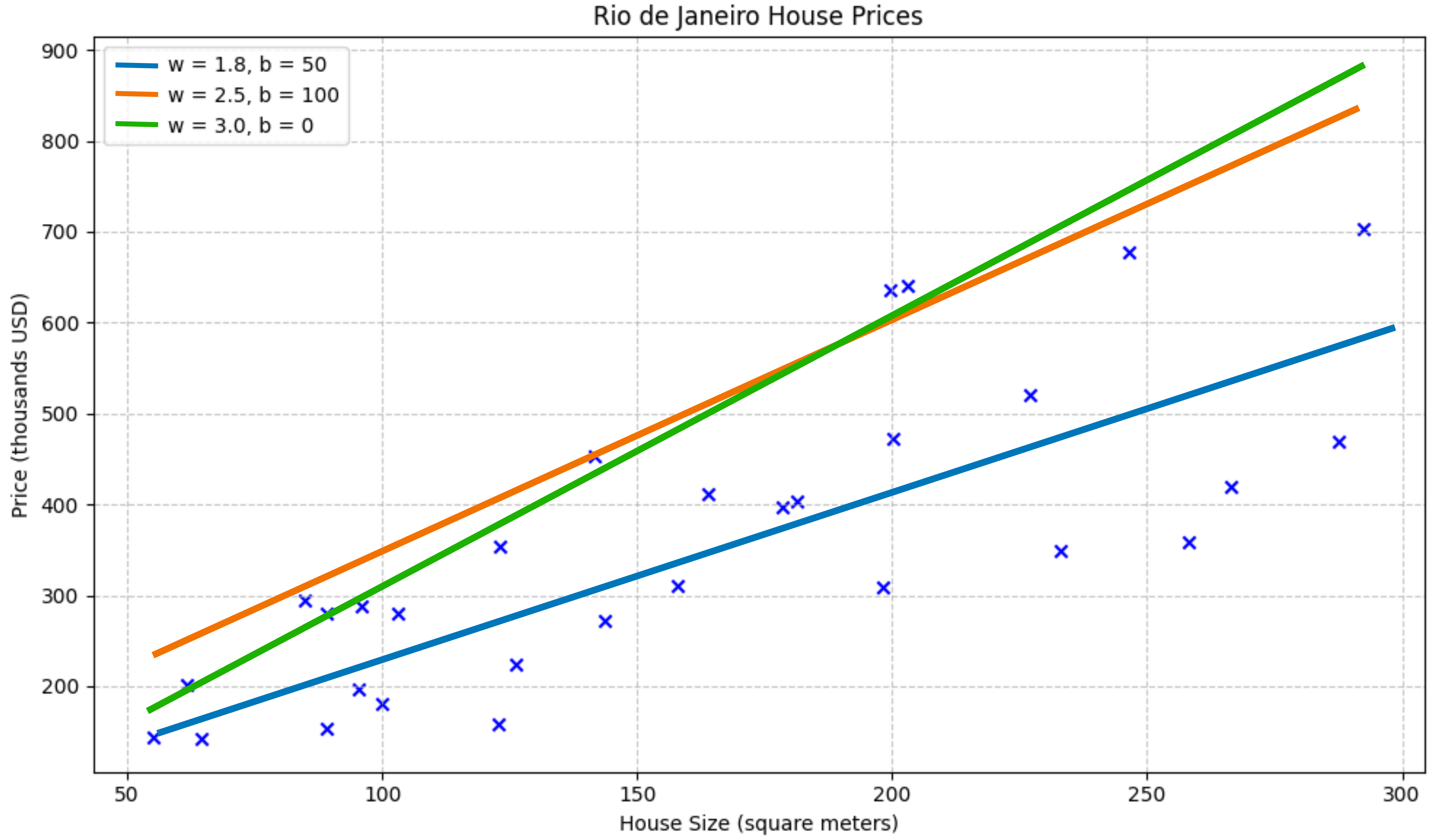
- **Convex function**

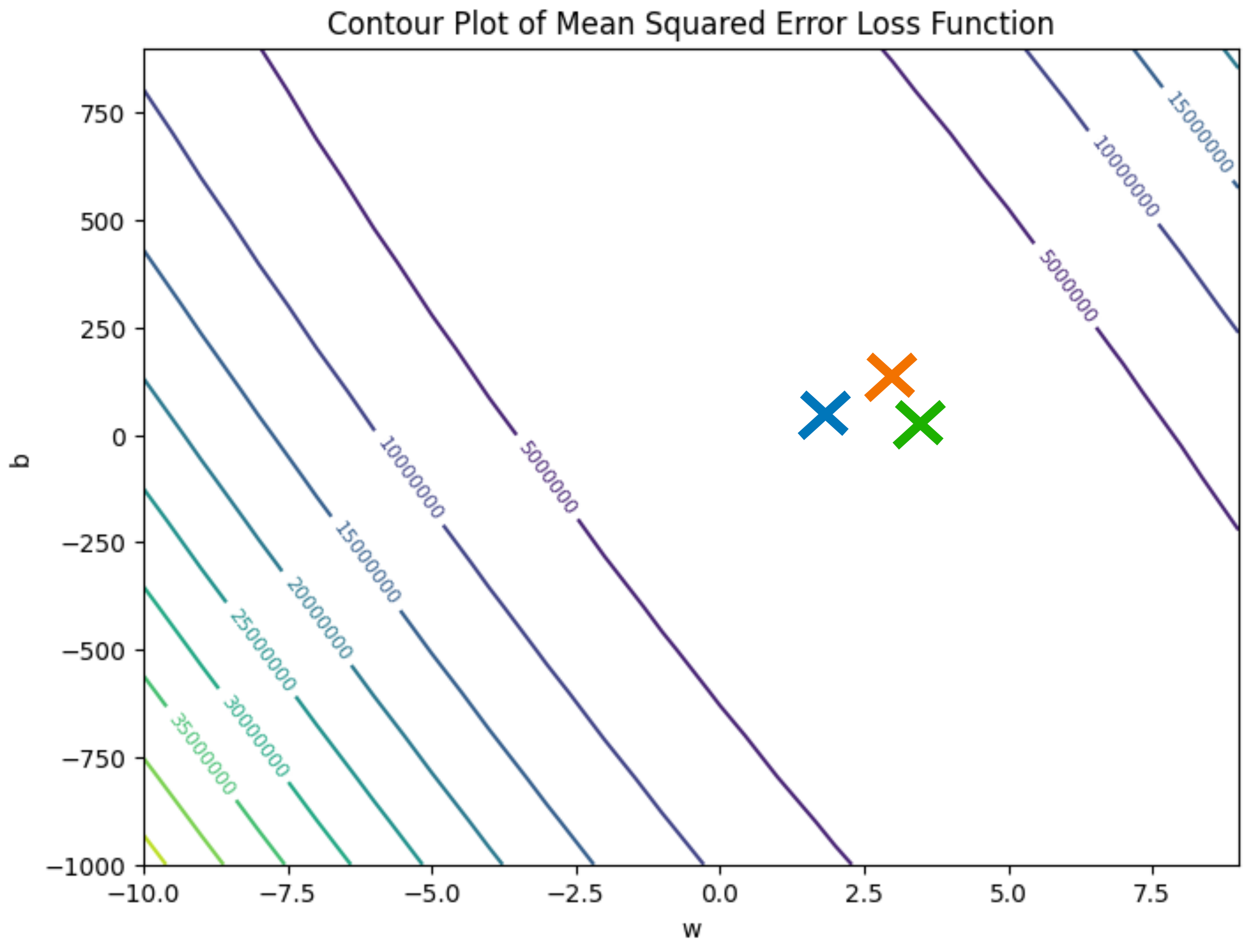
Only one (global) minimum!

Loss Function (complete)

- Complete hypothesis: $h(x) = wx + b$
- Loss function $L(h) = \frac{1}{2m} \sum_{i=1}^{n} (h(x^{(i)}) - y^{(i)})^2$



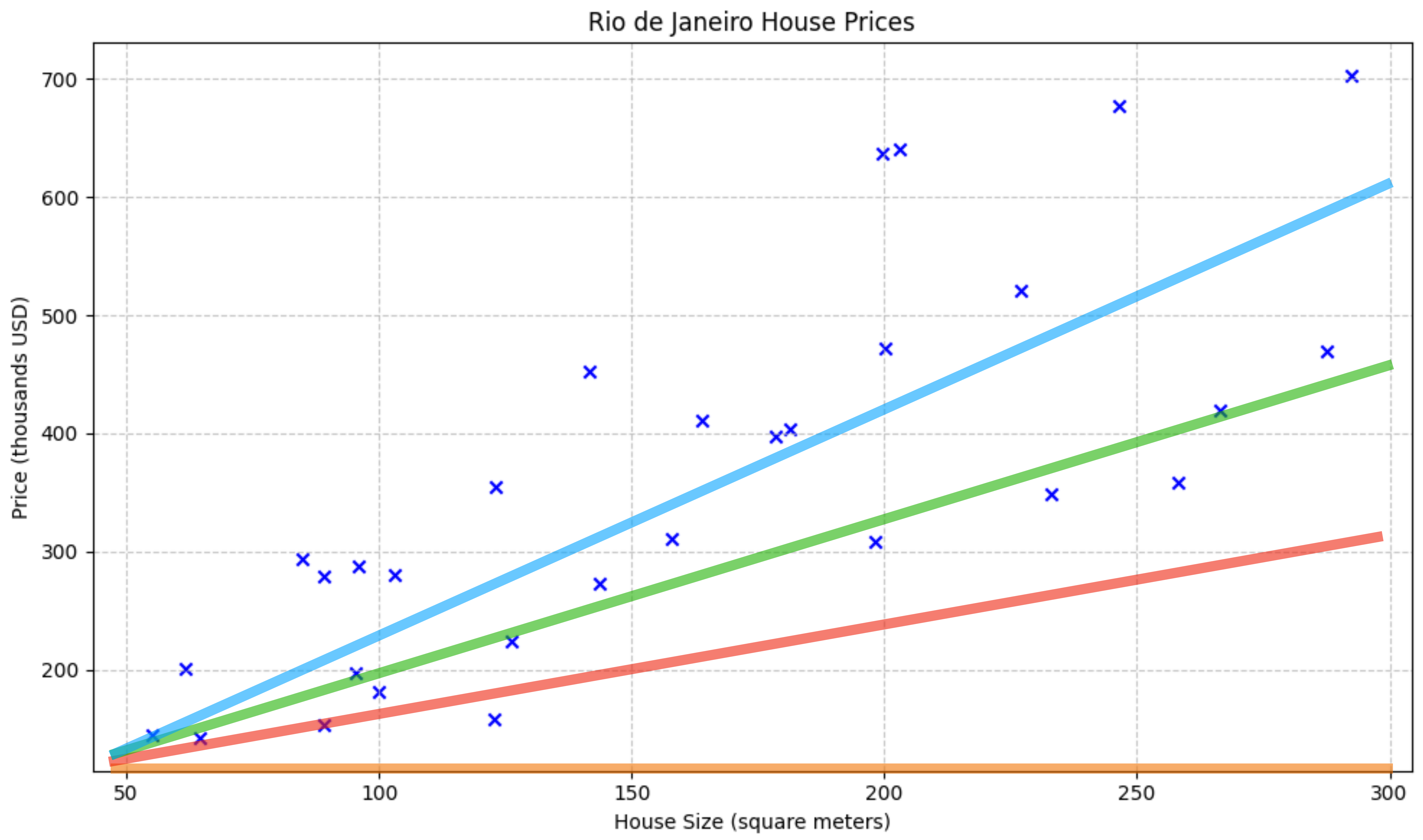


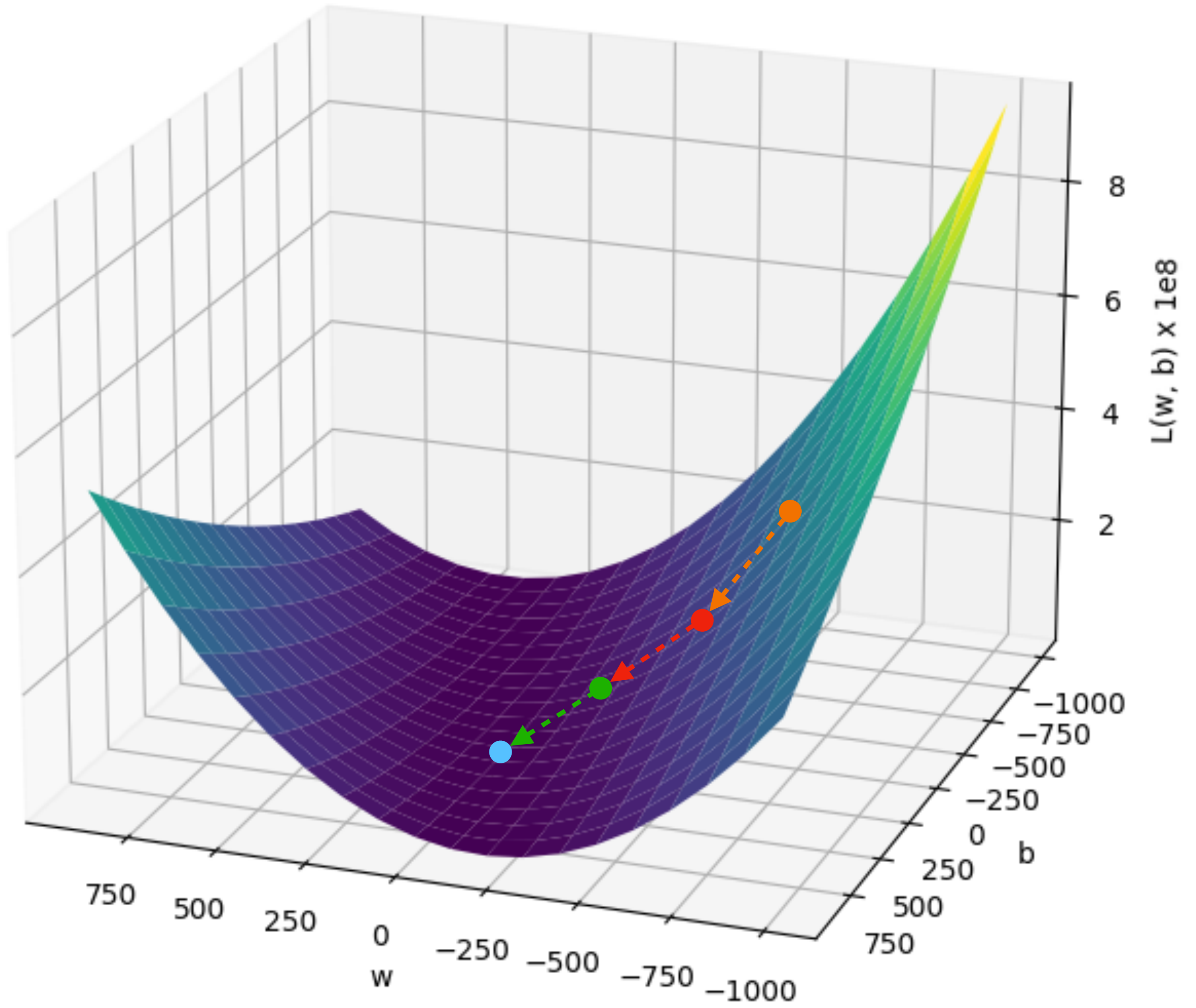


Gradient Descent

Start with given w, b values and iteratively update these values in the direction of steepest descent of L until we settle at or near a minimum

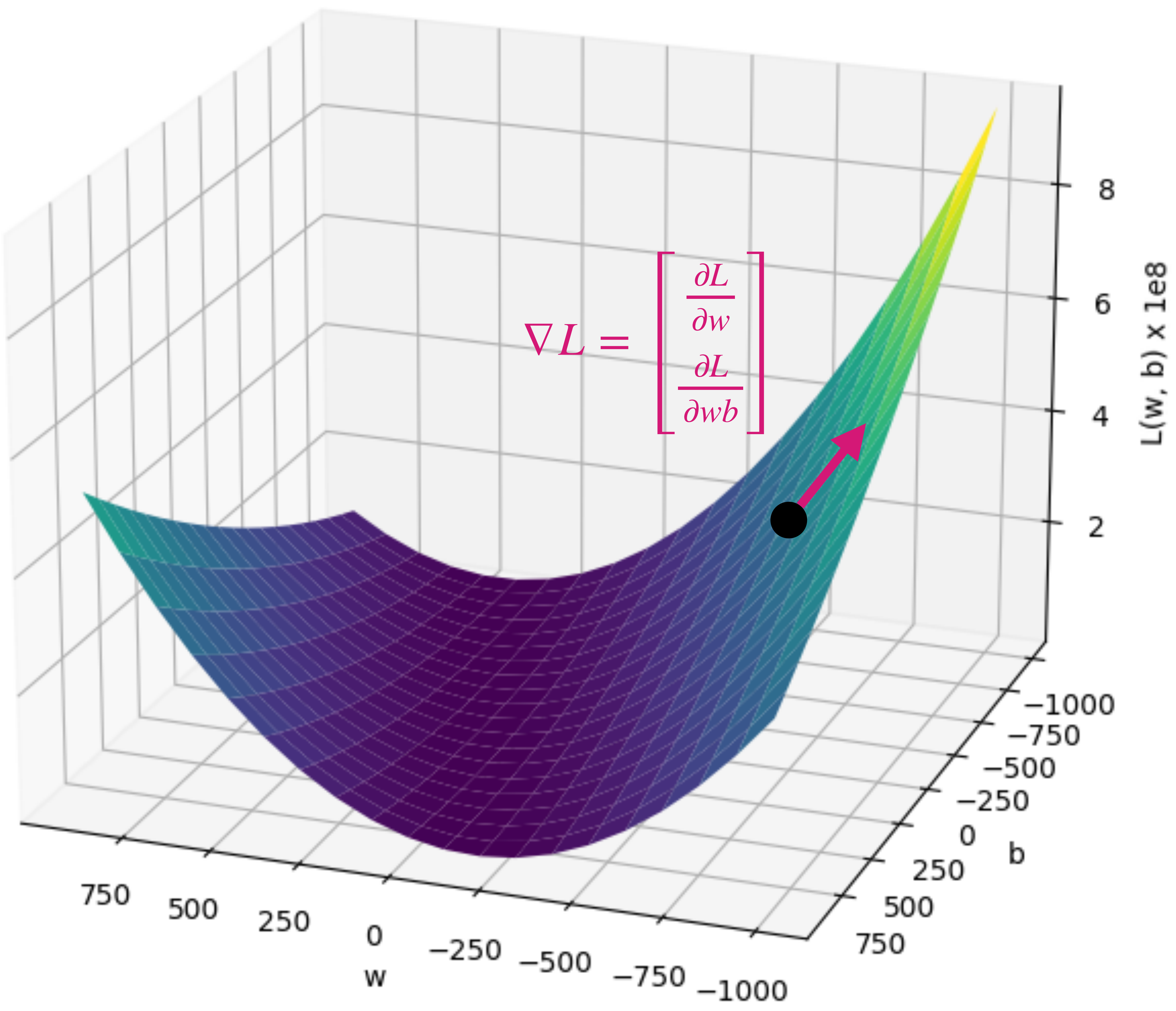
How to calculate the direction of movement? **Gradient vector!**





UFV

Gradient Vector



The **gradient vector** ∇L of a multivariate function $L(w_1, w_2, \ldots, w_d)$ is a vector where each element ∇L_i is the partial derivative of L with respect to w_i:

$$\nabla L = \begin{bmatrix} \frac{\partial L}{\partial w_1} \\ \frac{\partial L}{\partial w_2} \\ \vdots \\ \frac{\partial L}{\partial w_d} \end{bmatrix}$$

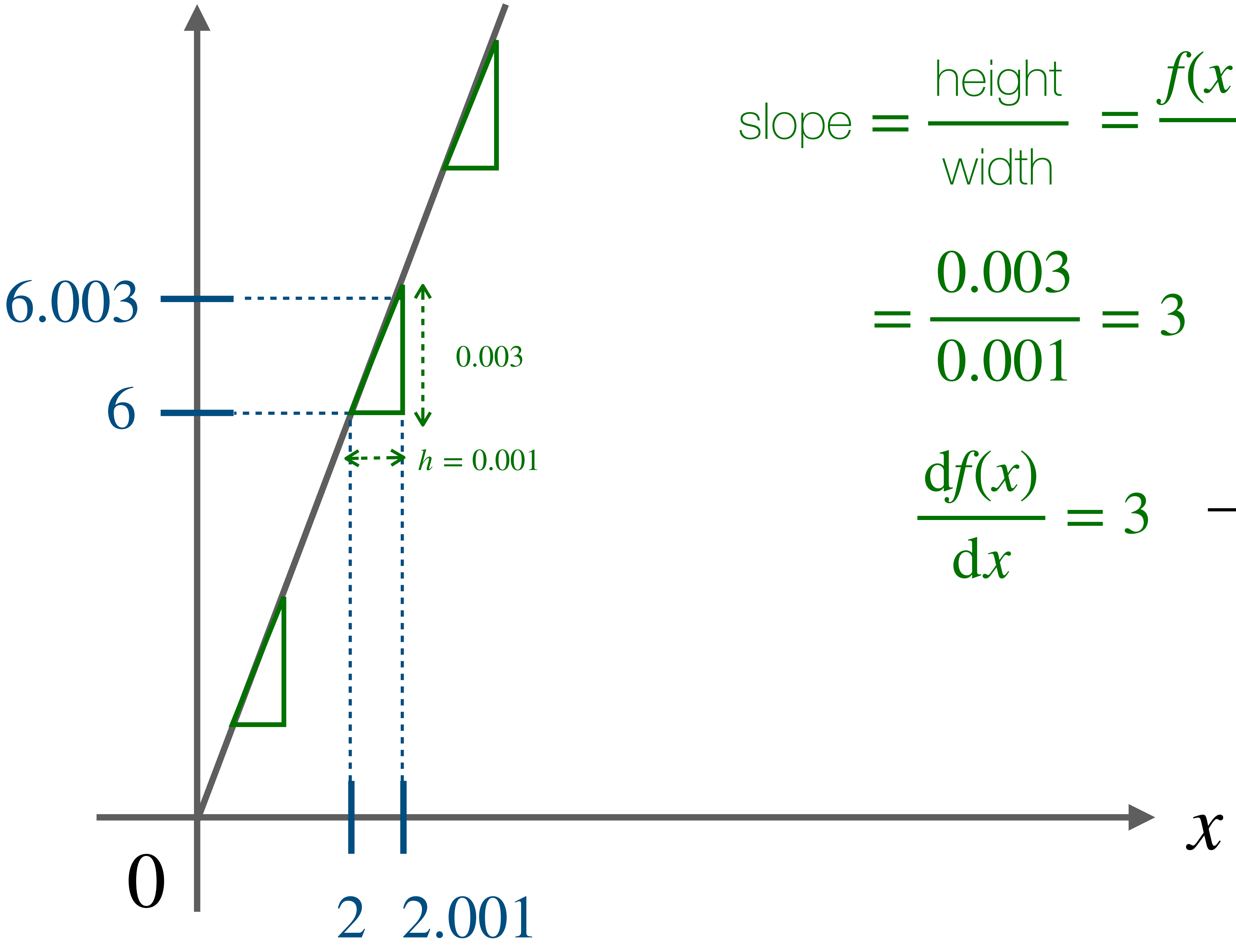
The vector $\nabla L(w)$ points to the direction of fastest increase of L at point w.

UFV

Derivatives

The derivative of a function L at the point $x = a$ represents the **slope** of the tangent line to that function at the point a

$$f(x) = 3x$$



$$\text{slope} = \frac{\text{height}}{\text{width}} = \frac{f(x+h) - f(x)}{h}$$

$$= \frac{0.003}{0.001} = 3$$

$$\frac{\mathrm{d}f(x)}{\mathrm{d}x} = 3 \longrightarrow$$ The derivative of $f(x)$ at $x = 2$ is 3

$h = 0.001$

$x = 2$	$f(x) = 6$
$x + h = 2.001$	$f(x + h) = 6.003$
$x = 5$	$f(x) = 15$
$x + h = 5.001$	$f(x + h) = 15.003$

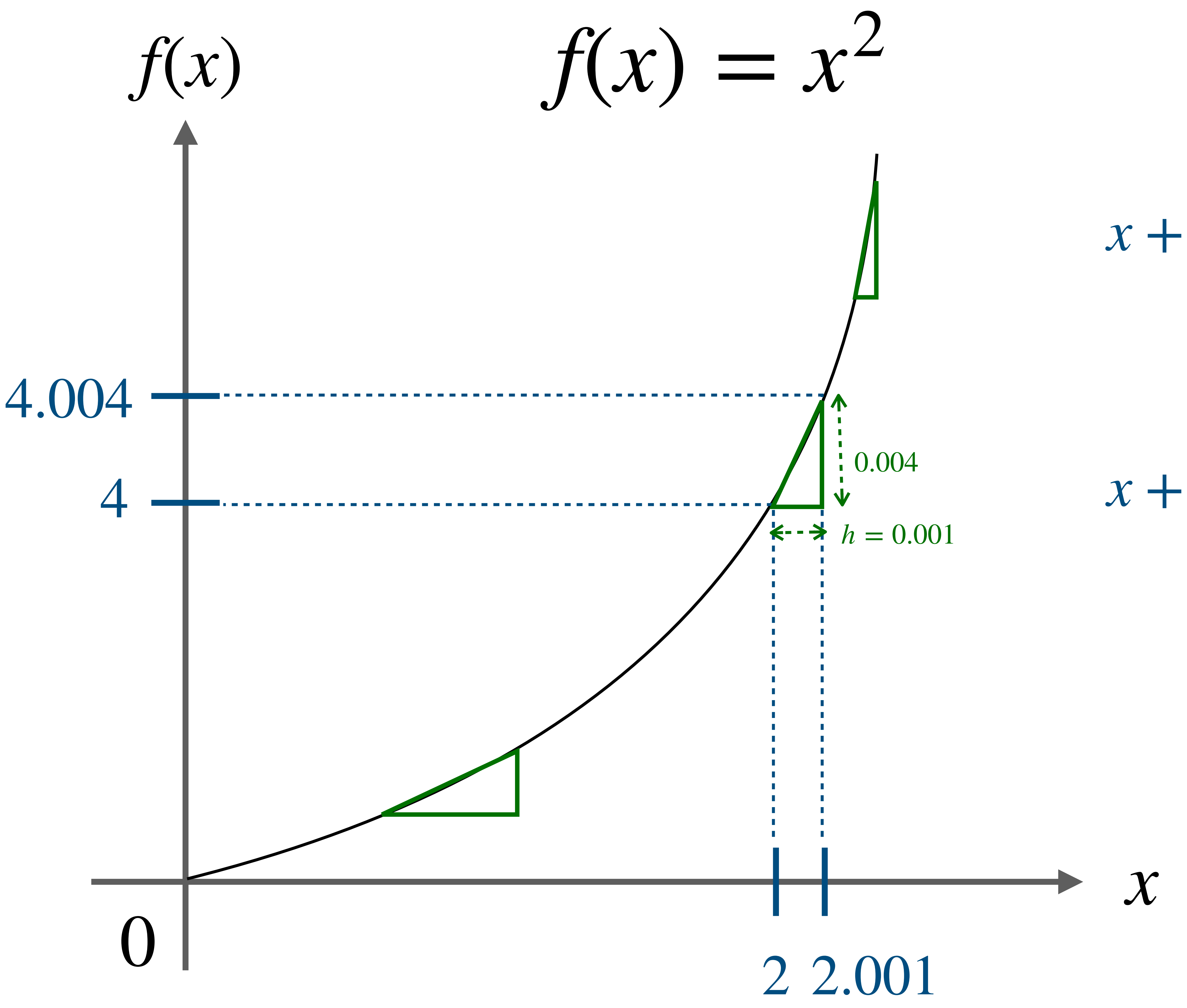
How much $f(x)$ is affected when we add a tiny variation uma h to x.

In Calculus, this variation h is infinitely small:

$$\frac{\mathrm{d}f(x)}{\mathrm{d}x} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Derivatives

The derivative of a function L at the point $x = a$ represents the **slope** of the tangent line to that function at the point a



$h = 0.001$

$x = 2$ $\quad f(x) = 4$

$x + h = 2.001$ $\quad f(x+h) \approx 4.004$

$$\frac{\mathrm{d}f(x)}{\mathrm{d}x} = \frac{0.004}{0.001} = 4$$

$x = 5$ $\quad f(x) = 25$

$x + h = 5.001$ $\quad f(x+h) = 25.010$

$$\frac{\mathrm{d}f(x)}{\mathrm{d}x} = \frac{0.0010}{0.001} = 10$$

$$\frac{\mathrm{d}f(x)}{\mathrm{d}x} = \frac{\mathrm{d}x^2}{\mathrm{d}x} = 2x$$

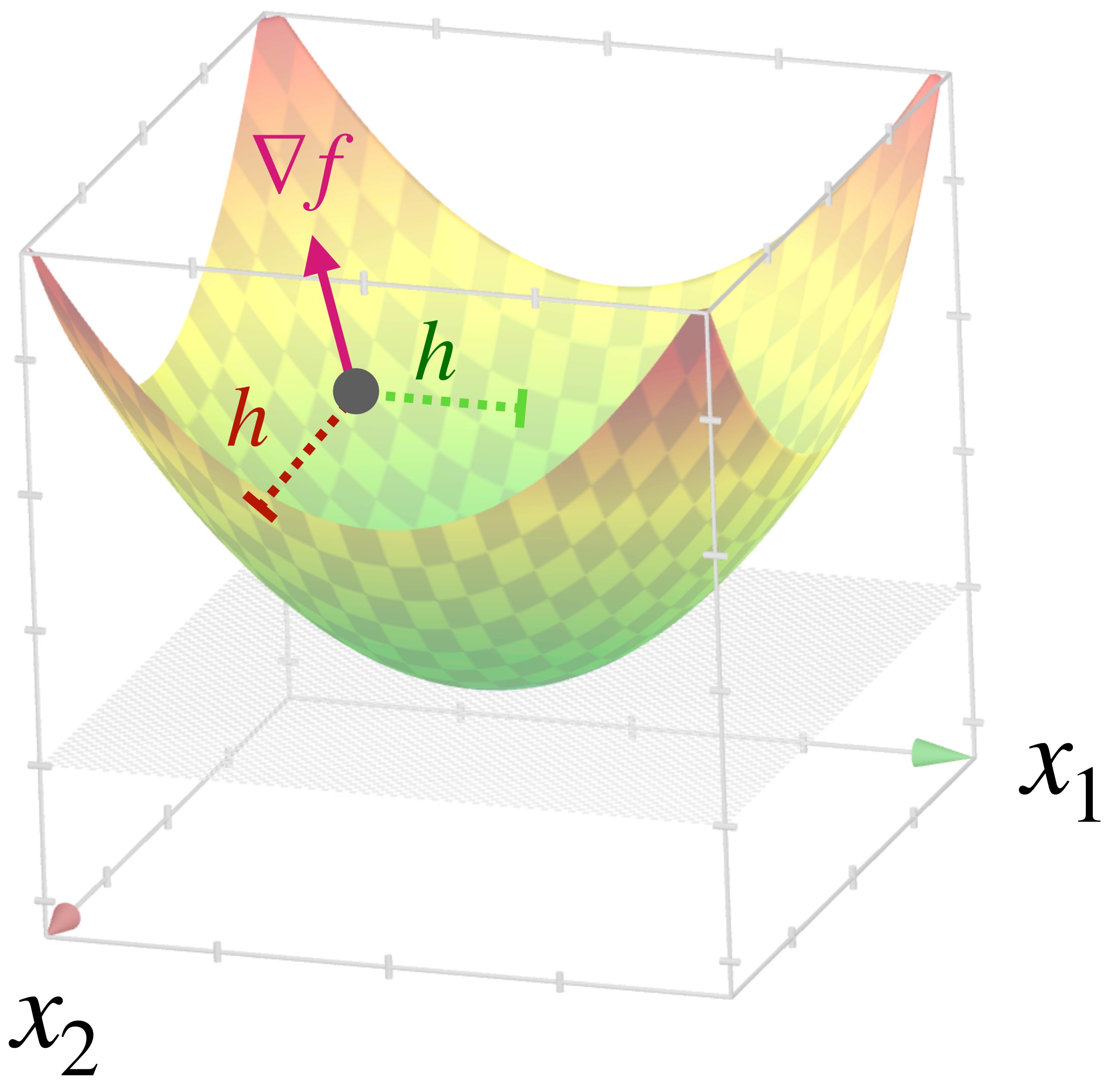
Derivative Rules

1. **Constant Rule:**
$$\frac{d}{dx}(c) = 0$$
2. **Constant Multiple Rule:**
$$\frac{d}{dx}[cf(x)] = cf'(x)$$
3. **Power Rule:**
$$\frac{d}{dx}(x^n) = nx^{n-1}$$
4. **Sum Rule:**
$$\frac{d}{dx}[f(x) + g(x)] = f'(x) + g'(x)$$
5. **Difference Rule:**
$$\frac{d}{dx}[f(x) - g(x)] = f'(x) - g'(x)$$
6. **Product Rule:**
$$\frac{d}{dx}[f(x)g(x)] = f(x)g'(x) + g(x)f'(x)$$
7. **Quotient Rule:**
$$\frac{d}{dx}\left[\frac{f(x)}{g(x)}\right] = \frac{g(x)f'(x) - f(x)g'(x)}{[g(x)]^2}$$
8. **Chain Rule:**
$$\frac{d}{dx}[f(g(x))] = f'(g(x))g'(x)$$

Partial Derivatives

The **partial derivative** of a multivariate function $f(x_1, x_2, \ldots, x_d)$ is its derivative with respect to one of its variables x_i, and represents the rate of change of the function in the x_i-direction.

$$f(x_1, x_2) = x_1^2 + x_2^2$$



$$(x_1, x_2) = (2,\ 5)$$

$$\frac{\partial f(x_1, x_2)}{\partial x_1} = \frac{\partial x_1^2}{\partial x_1} + \frac{\partial x_2^2}{\partial x_1} = 2x_1 + 0 \ = 2x_1 \ = 2 \times 2 \ = 4$$

$$\frac{\partial f(x_1, x_2)}{\partial x_2} = \frac{\partial x_1^2}{\partial x_2} + \frac{\partial x_2^2}{\partial x_2} = 0 + 2x_2 \ = 2x_2 \ = 2 \times 5 \ = 10$$

The gradient vector $\nabla f(x_1, x_2)$ is defined by the partial derivatives of $f(x_1, x_2)$

$$\nabla f(x_1, x_2) = \begin{bmatrix} \frac{\partial f(x_1, x_2)}{\partial x_1} \\ \frac{\partial f(x_1, x_2)}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2x_1 \\ 2x_2 \end{bmatrix} = \begin{bmatrix} 4 \\ 10 \end{bmatrix}$$

UFV

Chain rule

$$f(x) = (x^2 + 1)^3$$

Internal function:

$$g(x) = x^2 + 1 \qquad \frac{\mathrm{d}g}{\mathrm{d}x} = 2x$$

External function:

$$f(g(x)) = g(x)^3 \qquad \frac{\mathrm{d}f}{\mathrm{d}g} = 3(g(x))^2$$

$$\frac{\mathrm{d}f}{\mathrm{d}x} = \frac{\mathrm{d}f}{\mathrm{d}g} \cdot \frac{\mathrm{d}g}{\mathrm{d}x} = 3(x^2 + 1)^2 \cdot (2x) = 6x(x^2 + 1)^2$$

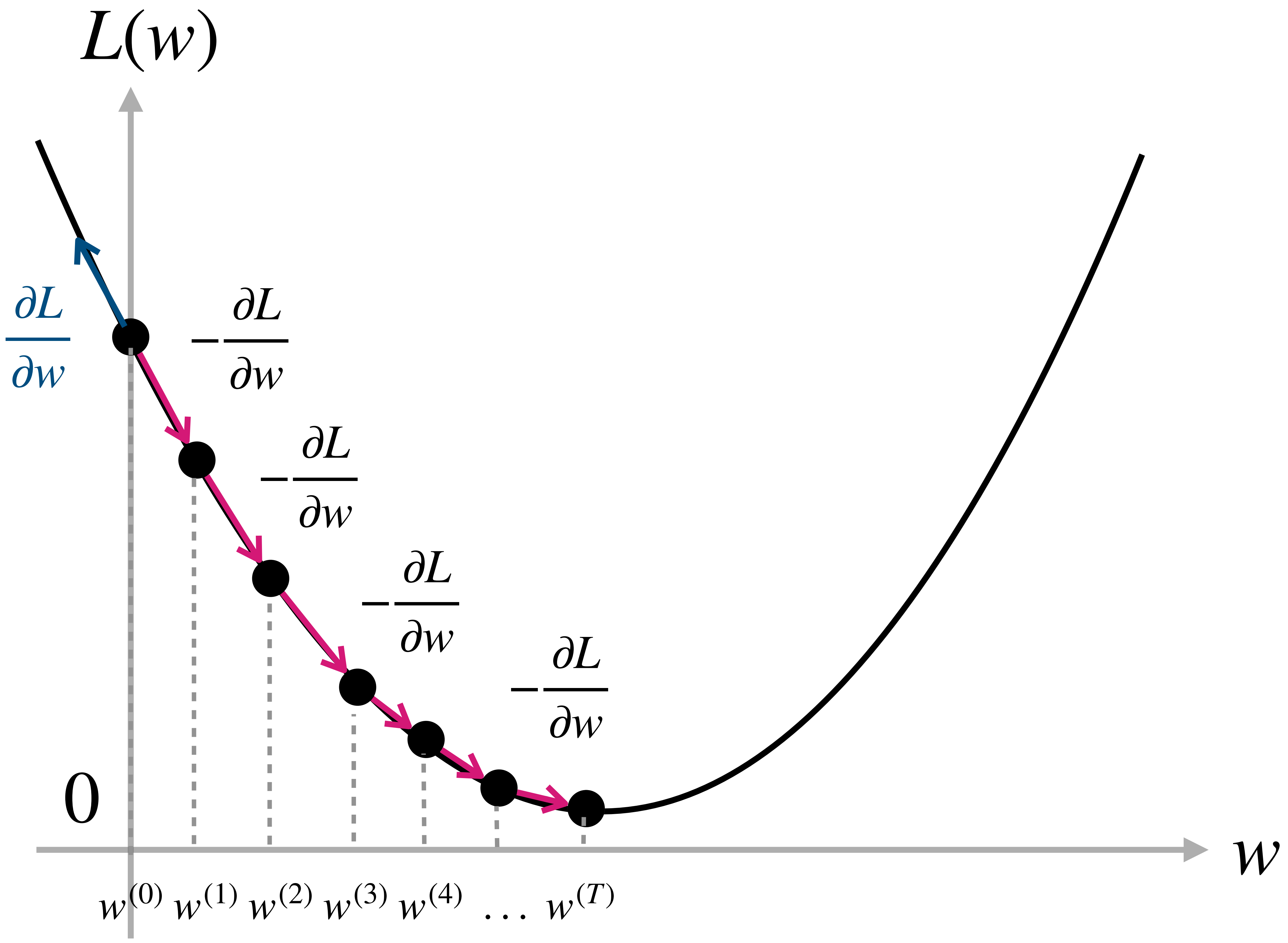
To calculate the derivative of composite function $f(g(x))$, we must use the **chain rule:**

$$\frac{\mathrm{d}f}{\mathrm{d}x} = \frac{\mathrm{d}f}{\mathrm{d}g} \cdot \frac{\mathrm{d}g}{\mathrm{d}x}$$

The derivative of the composite function $f(g(x))$ is the product of the derivative of the external function f with respect to g by the derivative of the internal function g with respect to x.

UFV

Gradient Descent



Start with given w, b values and iteratively update these values in the direction of steepest descent of L:

$$w_t \leftarrow w_{t-1} - \alpha \frac{\partial L}{\partial w}$$

$$b_t \leftarrow b_{t-1} - \alpha \frac{\partial L}{\partial b}$$

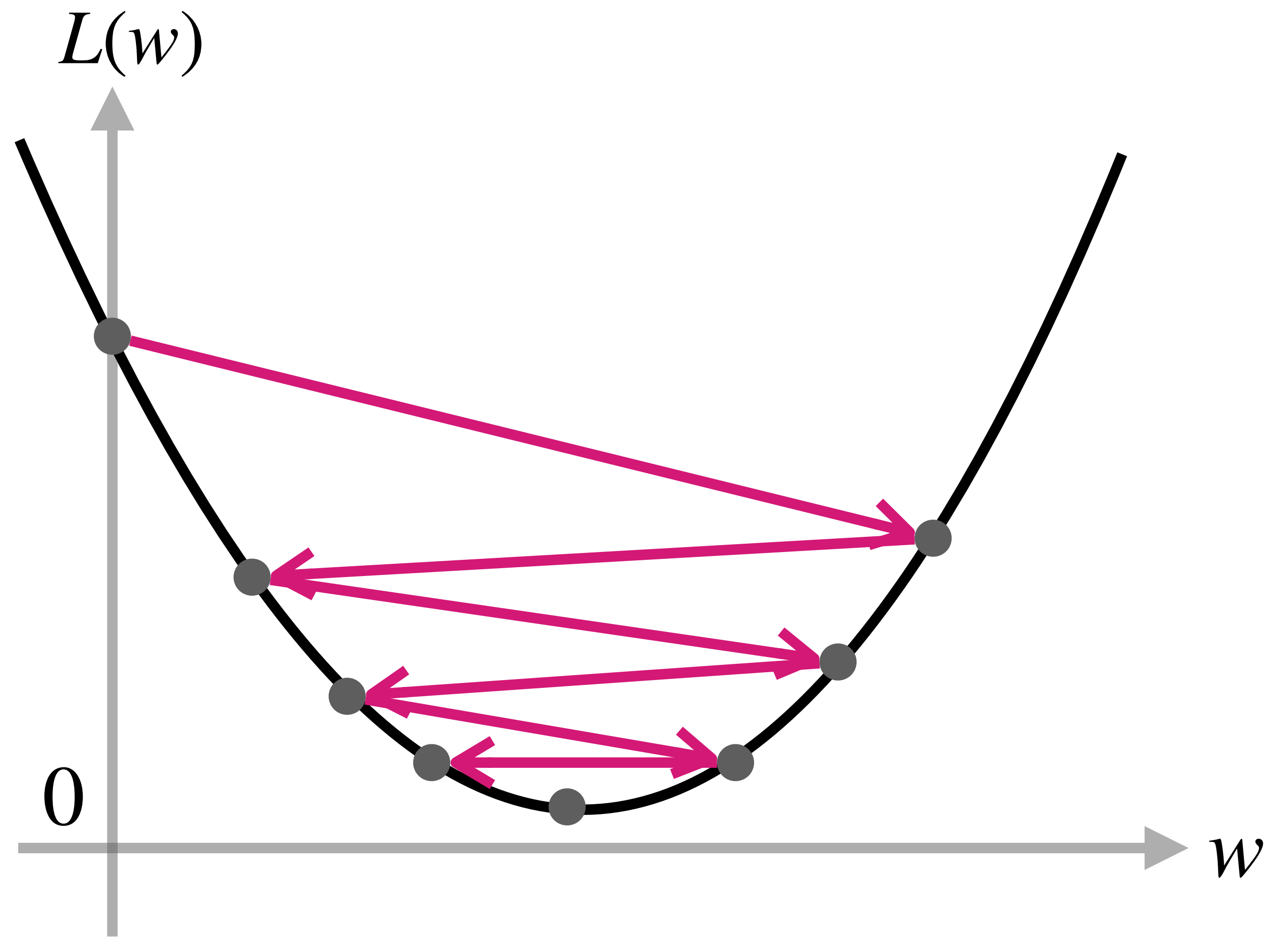
where α is a hiperparameter called **learning rate**, that controls the length of the gradient vector.

Learning Rate

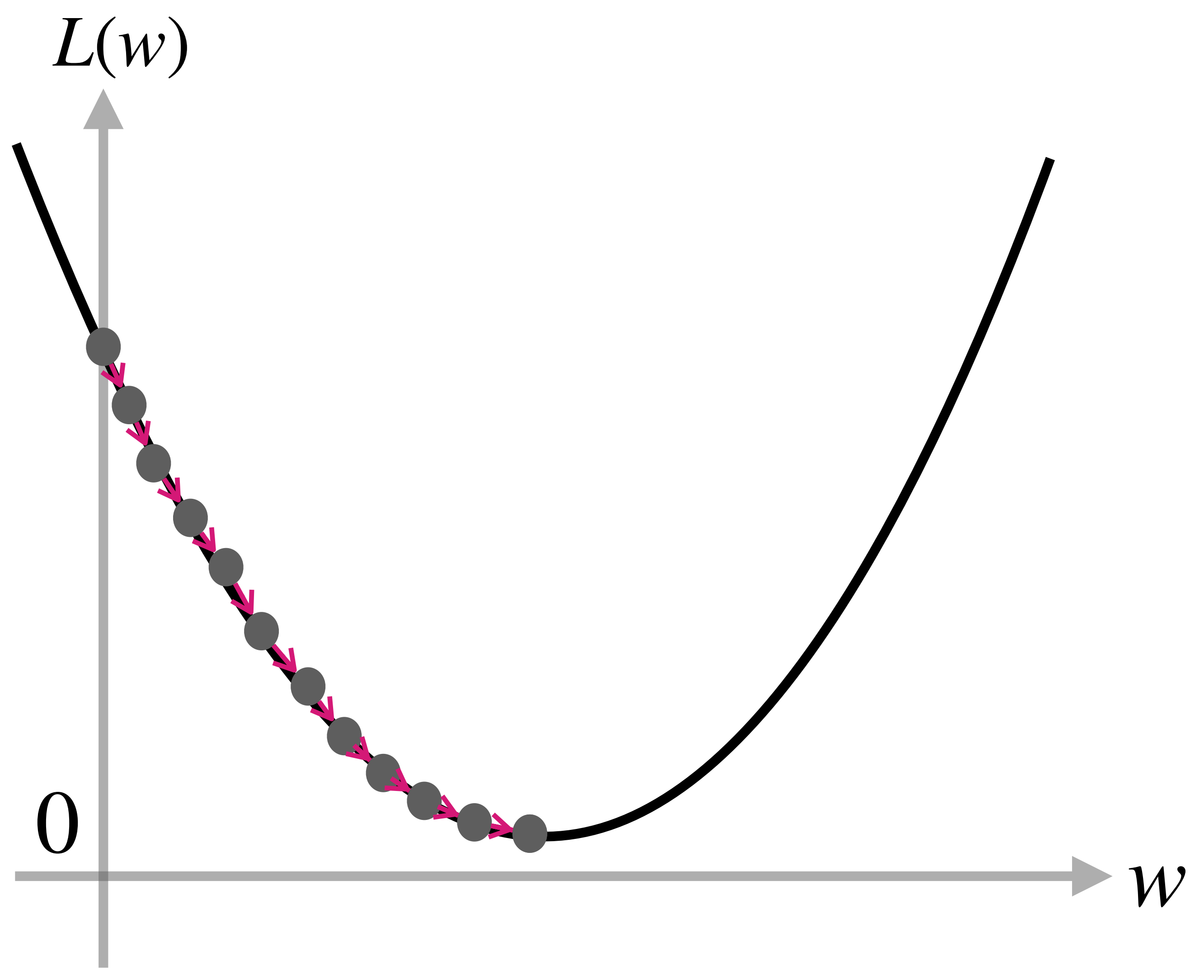
- **Gradient descent**

$$w_t \leftarrow w_{t-1} - \alpha \frac{\partial L}{\partial w}$$

$$b_t \leftarrow b_{t-1} - \alpha \frac{\partial L}{\partial b}$$



Large learning rate
Fast convergence, but suboptimal!



Small learning rate
Slow convergence and can get stuck in local minima!

UFV

Calculating the gradients for linear regression

$$\frac{\partial L}{\partial w} = \frac{\partial}{\partial w}\frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2 \;= \frac{\partial}{\partial w}\frac{1}{2m}\sum_{i=1}^{m}(wx^{(i)} + b - y^{(i)})^2 \;= \frac{\partial}{\partial w}\frac{1}{2m}\sum_{i=1}^{m}2\,(wx^{(i)} + b - y^{(i)})\cdot\frac{\partial}{\partial w}wx^{(i)} + b - y^{(i)}$$

$$= \frac{\partial}{\partial w}\frac{1}{2m}\sum_{i=1}^{m}2(wx^{(i)} + b - y^{(i)})\,x^{(i)} \;= \frac{1}{m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})\,x^{(i)}$$

$$\frac{\partial L}{\partial b} = \frac{\partial}{\partial b}\frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2 \;= \frac{\partial}{\partial b}\frac{1}{2m}\sum_{i=1}^{m}(wx^{(i)} + b - y^{(i)})^2 \;= \frac{\partial}{\partial b}\frac{1}{2m}\sum_{i=1}^{m}2\,(wx^{(i)} + b - y^{(i)})\cdot\frac{\partial}{\partial b}wx^{(i)} + b - y^{(i)}$$

$$= \frac{\partial}{\partial w}\frac{1}{2m}\sum_{i=1}^{m}2\,(wx^{(i)} + b - y^{(i)}) \;= \frac{1}{m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})$$

Gradient Descent for Linear Regression

```python
def optimize(x, y, lr, n_iter):
  # Init weights to zero
  w, b = 0, 0

  # Optimize weihts iteratively
  for t in range(n_iter):
    # Predict x labels with w and b
    y_hat = np.dot(w,x) + b

    # Compute gradients
    dw = (1 / m) * np.sum((y_hat - y) * x)
    db = (1 / m) * np.sum(y_hat - y)

    # Update weights
    w = w - lr * dw
    b = b - lr * db

  return w, b
```

Linear Regression

$$h(x) = wx + b$$

Loss function

$$L(h) = \frac{1}{2m} \sum_{i=1}^{n} (h(x^{(i)}) - y^{(i)})^2$$

Gradient

$$\frac{\partial L}{\partial w} = \frac{1}{m} \sum_{i=1}^{n} (h(x^{(i)}) - y^{(i)})x^{(i)}$$

$$\frac{\partial L}{\partial b} = \frac{1}{m} \sum_{i=1}^{n} (h(x^{(i)}) - y^{(i)})$$

Next Lecture

L4: Logistic Regression

A linear model for linearly separable classification problems