Gradient Descent Optimization

Navigating the Loss Landscape

Sina Moghimi, Autumn 2024

Outline

- Basic Optimization Concept
- Gradient Descent Fundamentals
- Learning Rate Dynamics
- Stochastic Gradient Descent (SGD)
- Batch Gradient Descent
- Convergence Strategies
- Advanced Optimization Techniques
- Practical Considerations

What is Optimization?

- **Optimization** is the procedure
    - Used to improve a model's performance
        - By finding the best set of parameters (weights and biases)
- **Goal**
    - Minimize the loss function
        - Ensures that the model maps inputs to outputs as accurate as possible
- **Intuition**
    - Imagine a **landscape**(loss function) with hills and valleys
        - We are interested in valleys
        - The optimization process takes us step by step closer to one of the valleys
        - Depending on where we are, we go to the nearest valley
        - Step size (learning rate) would be the parameter defining our speed to get to the bottom asap
        - We need a direction as well!
    - Imagine we write a program to tell us if we should move to right or left in order to get to the bottom of the valley, what should we consider? The computer doesn't see the world as we do!

Gradient

- In place we are, we have access to the surrounding area, thus we can:

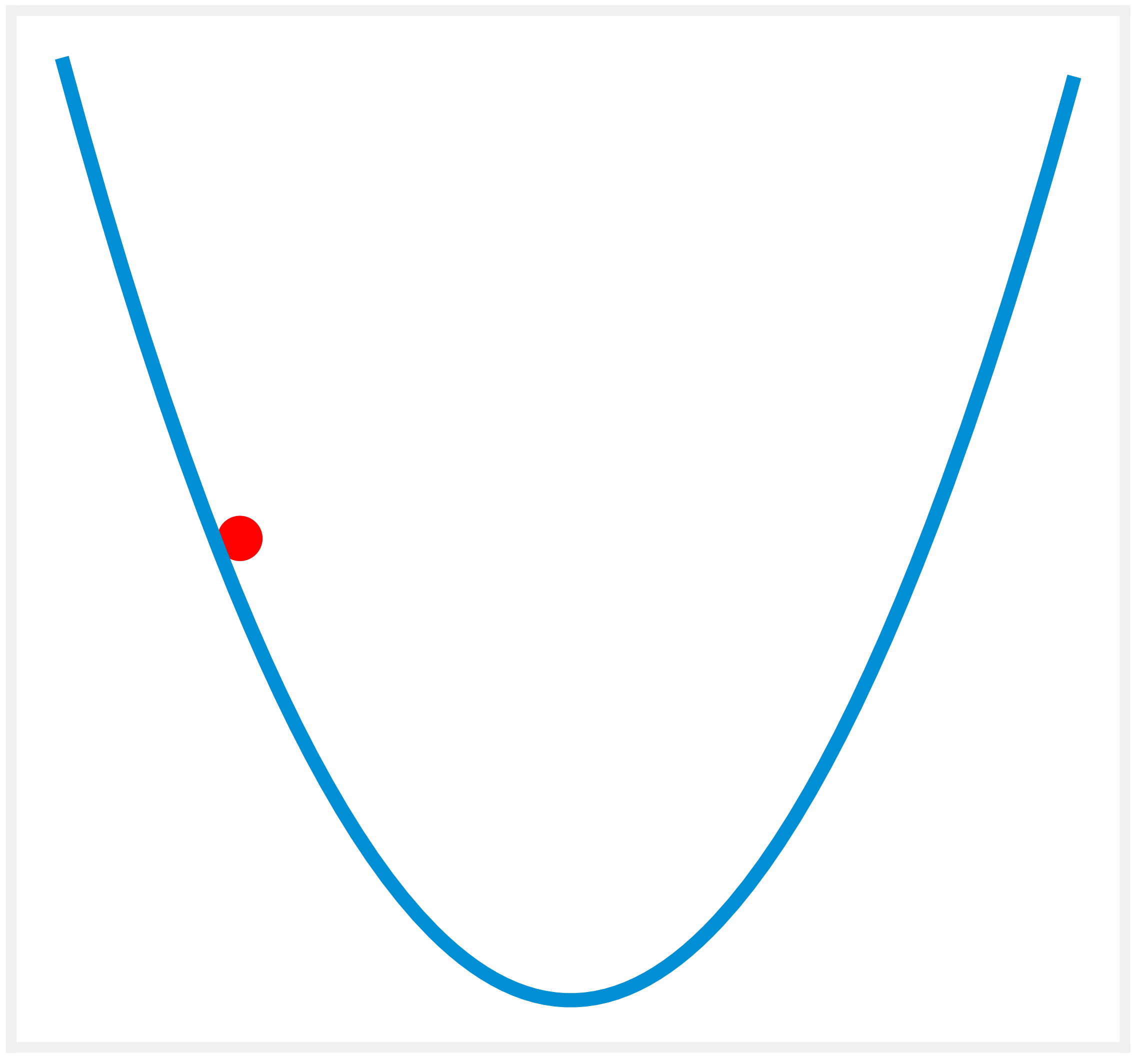
$$\frac{dy}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

- The derivative tells us if we are moving towards minimum or maximum

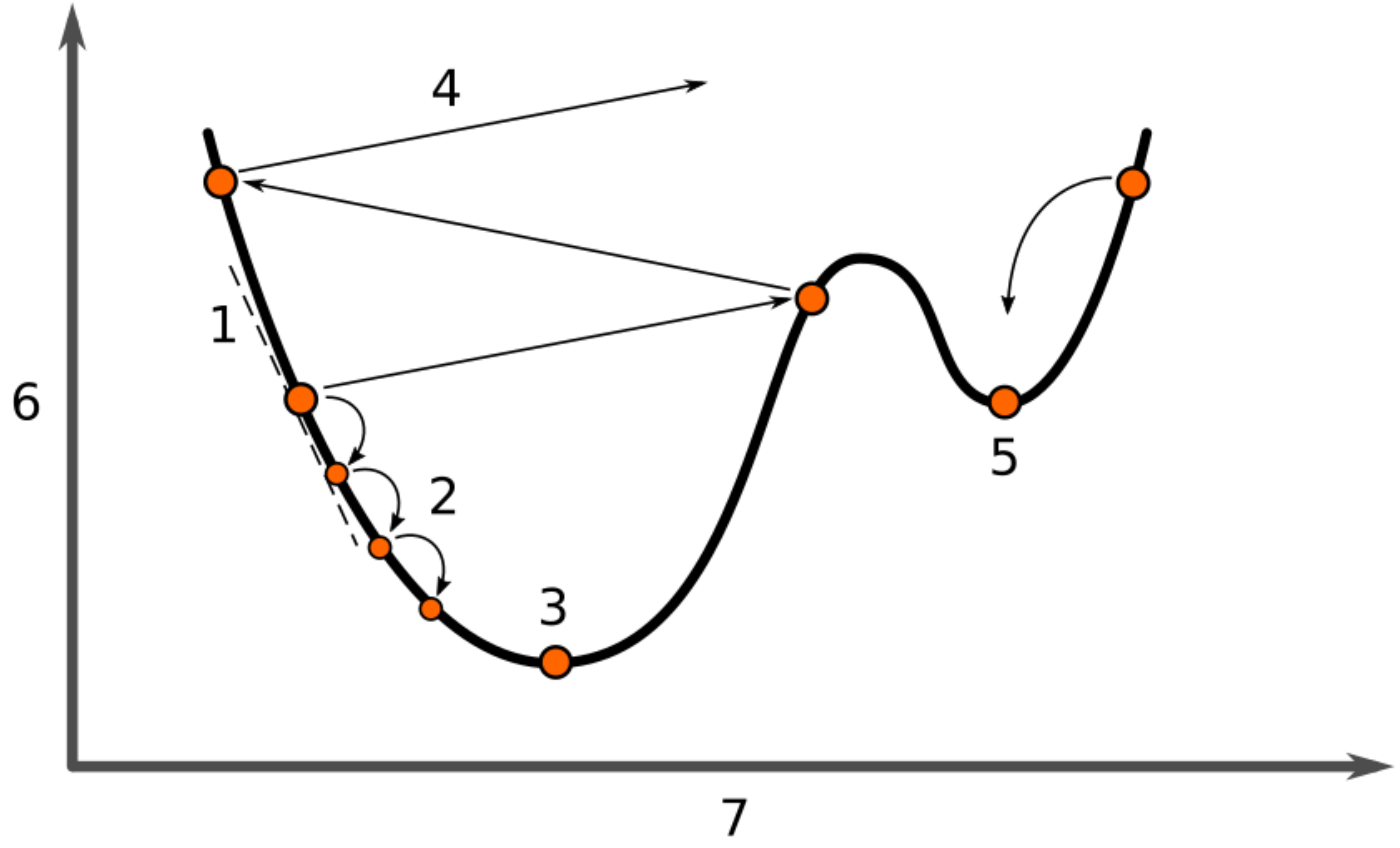
$$x_{new} = x - \frac{dy}{dx}$$

- What if the derivative is large? Can we move that much?

- Why we subtract?



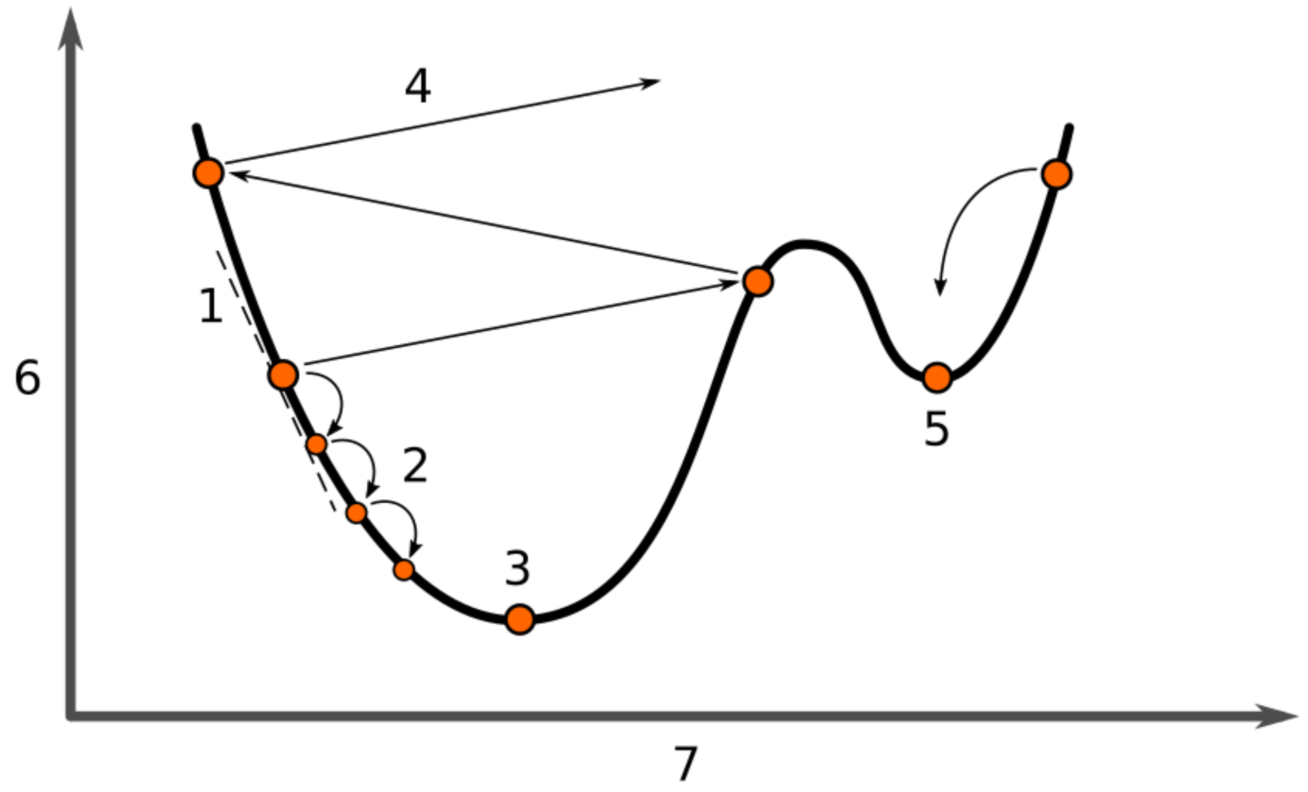
Gradient



Gradient

$$x_{new} = x - \eta \cdot \frac{dy}{dx}$$

- Should η be constant?!



Gradient Descent

- Gradient Decent is the **iterative** algorithm which uses the **gradient** to get to the **local minimum**
- Assume we have a vector of weights

$$[\omega_1, \omega_2, \omega_3, \ \ldots, \omega_n]$$

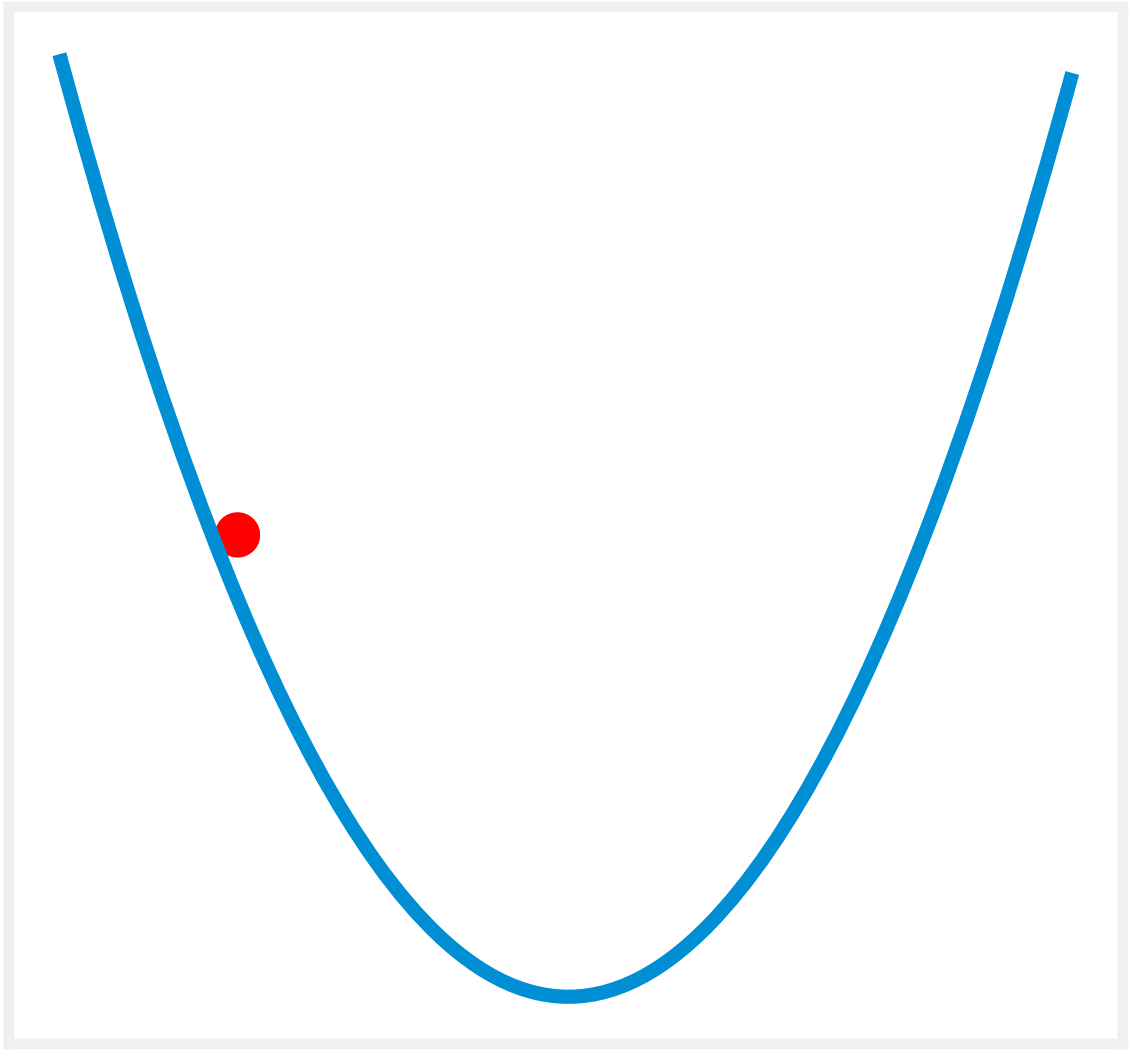
- The gradient of the loss function would be

$$\nabla L(\omega) = [\frac{\partial L}{\partial \omega_1}, \frac{\partial L}{\partial \omega_2}, \frac{\partial L}{\partial \omega_3}, \ \ldots, \frac{\partial L}{\partial \omega_n}]$$

- To update the parameters

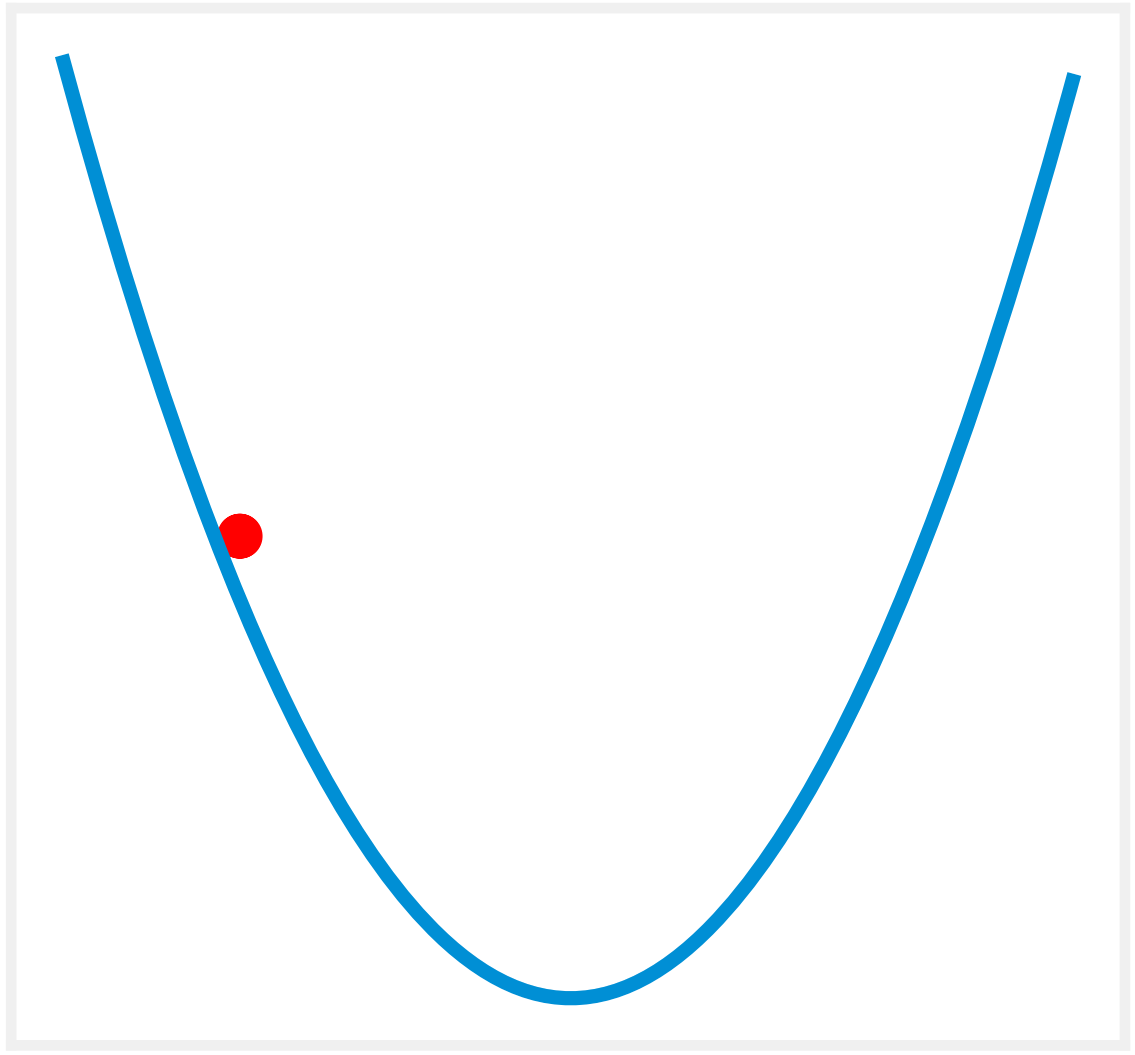
$$\omega = \omega - \eta \,.\, \nabla L(\omega)$$

- η is the learning rate, which controls the step size.
- $\nabla L(\omega)$ is the gradient of the loss function



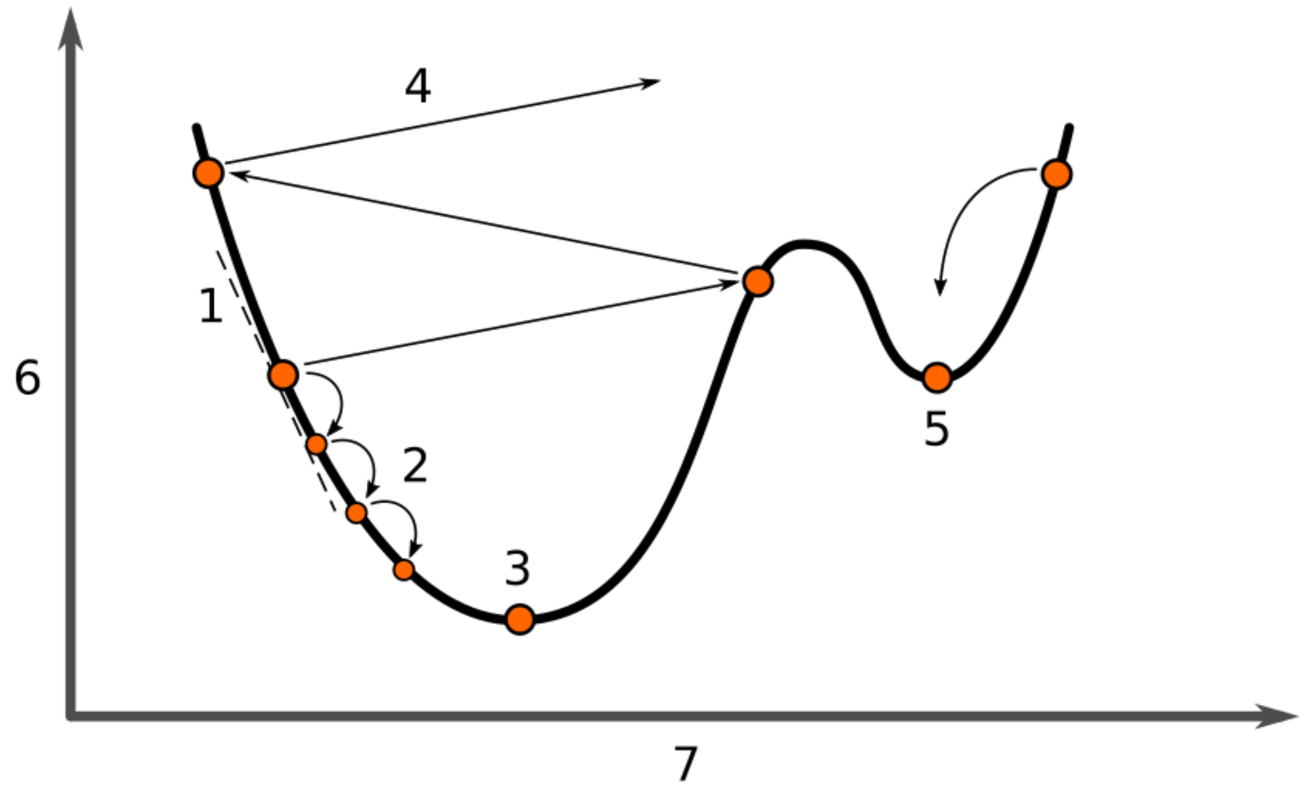
Gradient Descent

- To get to the minimum, we got to move against the slope
  - Think of a hiker on a hill trying to reach the lowest valley
  - The gradient represents the slope of the hill
  - By stepping downhill (opposite the gradient), the hiker moves closer to the valley



Optimization process

1. Initialize the Parameters
2. Compute the Error
3. Calculate Loss
4. Compute gradients
5. Update Parameters
6. Repeat



Variants of Gradient Descent

- **Batch Gradient Descent**
  - Uses the entire dataset
    - **Pros**: Accurate Gradients
    - **Cons**: Slow for large Datasets
- **Stochastic Gradient Descent (SGD)**
  - Uses a single sample at a time
    - **Pros**: Fast Update
    - **Cons**: High variance in updates, Instability
- **Mini-Batch Gradient Descent**
  - Uses a small subset of data
    - **Pros**: Balances the efficiency
    - **Cons**: None
-