Deep Learning Labwork 1

2440048 - Thien An NGUYEN

April 2025

1 Introduction

In this Labwork, we focus on Gradient Descent, where I implemented the algorithm in Python from scratch.
Gradient Descent is very straightforward, I need to calculate the Gradient of a Function and use the Gradient to find its local (global) minimum (if it exists). Gradient Descent converges to a global minimum in convex functions, while for non-convex functions, it may find a local minimum or a saddle point. Gradient Descent requires the function to be differentiable, and it also depends on the learning rate. Also, if a function has minimums at infinity, Gradient Descent won't find it.
Of course, there are other ways to find the global (local) minimum of a function, but Gradient Descent finds it iteratively, which is better in high-dimensional or large-scale problems where exact solutions are too costly to compute.

2 Implementation

2.1 The code

```
def gradient_descent(f, x_0, h, L, max_iterations, threshold = 1e-6):
    fx = []
    x_s = []
    for i in range(max_iterations):
        derivative = (f(x_0 + h) - f(x_0 - h)) / (2*h)
        x_a = x_0 - L * derivative
        f_xa = f(x_a)
        fx.append(f_xa)
        x_s.append(x_a)
        print(f"Step {i + 1}: x = {x_a:.6f} | f(x) = {f(x_a):.2f}")

        if abs(f(x_a) - f(x_0)) < threshold:
            break
        x_0 = x_a
```

```
return x_0, fx, x_s
```

2.2 Explanation

My code takes the following inputs:

- f: The function that we want to find the minimum of.
- x_0: Our starting point.
- h: The step to calculate the gradient (since I used numeric gradient calculation)
- L: The learning rate (or the step we want to take follows the Gradient.
- max_iterations: (self explanatory)
- threshold: The threshold at which, if the changes after each step are not too big anymore, we stop.

First, the derivative is calculated, then x_a is an updated version of the position by taking a small step L following the derivative. This will iterate through till all the iterations are done or till the updates are too small.

3 Experiments

We will use the $f(x) = x^2$ function for the experiment and experiment with the learning rate of 0.1, 0.3, and 1.2.

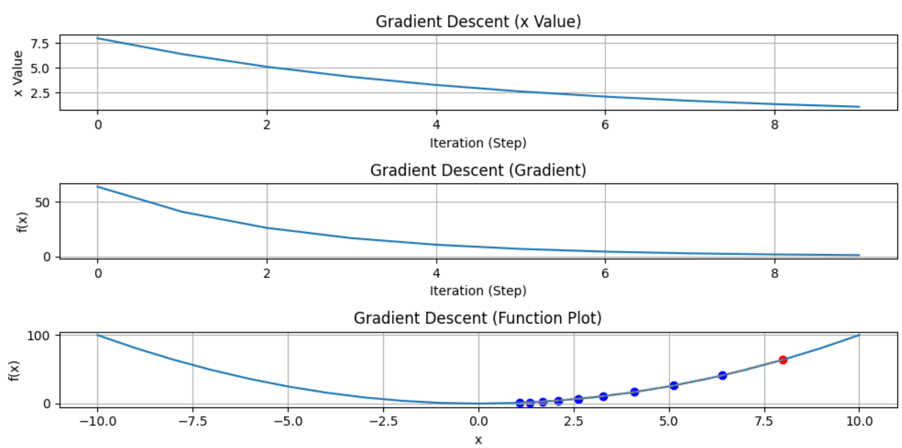

Figure 1: Gradient Descent with Learning Rate 0.1

Here, the red dot visualizes the position after step 1. We can see that Gradient Descent depends a lot learning rate. In the case of a learning rate of 0.1, it

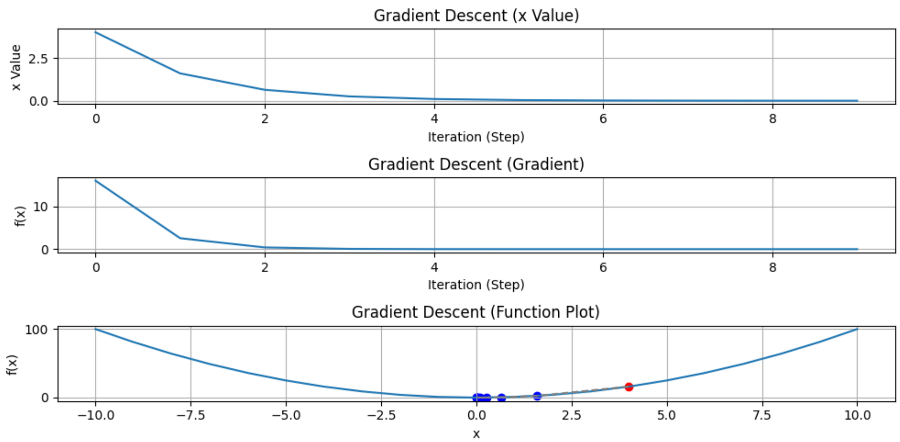

Figure 2: Gradient Descent with Learning Rate 0.3

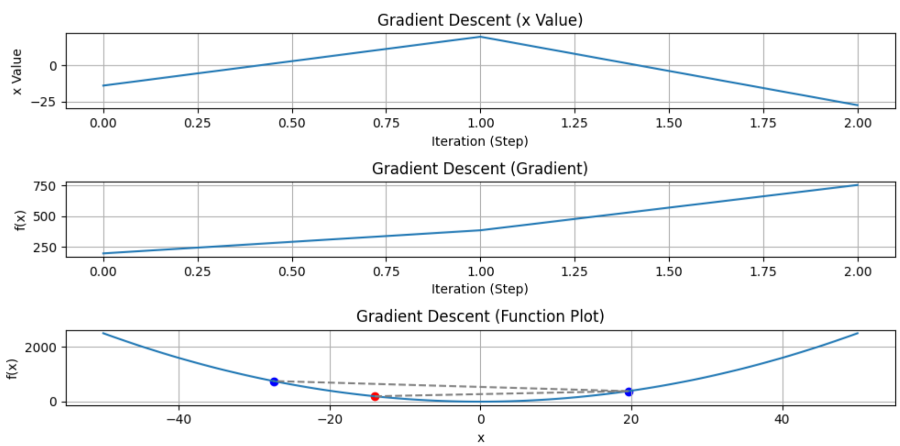

Figure 3: Gradient Descent with Learning Rate 1.2

converges, but it is a bit slow compared with a learning rate of 0.3. For a learning rate of 1.2, it diverges since it took too big of a step towards the minimum.
The gradient also decreased when the value is closer towards the minimum for the case of 0.1 and 0.3, but for 1,2 the gradient becomes larger cause of the divergence. At the eendnd x converges around 0.