Ain Shams University

Faculty of Engineering

Mechatronics Engineering Department

Computational Intelligence

Milestone (1)

Name	ID
Youssef Malak Samir	**1806173**
Pavly Ehab William	**1803239**
Bassam Sobhy Abd al Tawab	**1804313**

1. Problem definition

I. Optimization Problem

Choosing the best way to do something; Simply this is a definition to the optimization problem no matter how complex it is. In numerous fields optimization is used to get the shortest path to get a job done. In machine learning, optimization of learning parameters is used to get the best results out of the model used. Depending on a developed algorithm which finds minima of a function (the needed output) a model is applied to learn from training sets which help it reach its optimized parameters which make the model able to predict nearly perfect outputs.

In this report we will be discussing number of optimization algorithms; what are they? How do they work? how to implement them? as well real application and results to be able to compare different parameters value and their effect on a model's efficiency, also to compare between results of optimization resulting from different algorithms. .

2. Algorithms

I. Gradient descent

i. Species of gradient descent

- **Batch gradient descent:** The error for each point in each training set is added to the model once all training examples have been assessed. It's common to use the phrase "training phase." Even though this batching improves computing performance, large training datasets can take some time to evaluate because all the data must still be maintained in memory. Even though it frequently finds the local minimum rather than the global one, batch gradient descent frequently results in a continuous error gradient and convergence.
- **Mini-batch gradient descent:** Batch gradient descent and stochastic gradient descent are used in mini-batch gradient descent. Each of the smaller batches made from the training dataset receives updates. This method creates a balance between stochastic gradient descent's speed and the computing effectiveness of batch gradient descent.

ii. Implementation technique for gradient descent

We select moving in the gradient direction such that: - $f(X_0) \geq f(X_1) \geq f(X_2) \dots \geq f(X_{n-1}) \geq f(X_n)$

Equation: $X_{i+1} = X_i - \alpha \nabla f(x_i)$

Step 0: Select $X_0 \in R^n$, set α and i=0

Step 1: Compute $\nabla f(x_i)$ and H

Step 2: if $||\nabla f(x_i)|| < \varepsilon$, Stop, otherwise go to step 3

Step 3: Compute $X_{i+1} = X_i - \alpha \nabla f(x_i)$

Step 4: Update i = i + 1

Step 5: Go to Step 1

II. Newton Raphson Algorithm

a process that repeatedly uncovers the roots of a differentiable function F's solutions to the equation F(x) = 0. For a twice-differentiable function f, the roots of the derivative (also known as the crucial points of f, can be discovered using Newton's technique. The sections "Several variables" in Critical point (mathematics) and "Geometric interpretation" on this page have more details. These solutions could be saddle points, minima, or maxima. This is significant for optimization since it seeks to locate the (global) minima of the function f.

When it works, Newton's method is surprisingly simple to apply. It sometimes fails for a variety of reasons. when for some xi in the iterative process, f′(xi)=0.

i. When couldn't we apply Newton's

Consider the first error from the earlier section: Newton's method is unable to identify the precise spot where the tangent line intersects the x-axis when f(xi)=0 because the graph of f has a horizontal tangent line. The second failure is quite interesting mathematically even if it is not explored by this inquiry. It provides access to "dynamical systems theory" and "chaos theory," Recalling the prerequisites for the convergence of the Newtonian method: If f is continuously differentiable over U and f′(x) 0 for every x in U, you may demonstrate that for a function f, there exists an open interval UR1 around a root of f such that Newton's method converges for any x0 in U. Essentially, convergence can happen if f is "enjoyable" and U is too small

ii. Implementation technique for gradient descent

- Equation : $X_{i+1} = X_i - H^{-1} \nabla f(x_i)$

Step 0: Select X_0 E R^n, set α and i=0

Step 1: Compute $\nabla f(x_i)$ and H

Step 2: if $\|\nabla f(x_i)\| < \varepsilon$, Stop

Otherwise go to step 3

Step 3: Compute $\alpha = H^{-1}$

Step 4: Compute $X_{i+1} = X_i - \alpha \nabla f(x_i)$

Step 5: Update i = i + 1

Step 6: Go to Step 1

III. Steepest descent

The only way it differs from gradient descent is that we don't start with the assumption that (α), but rather go and get a value that has been minimised by one of the algorithms and use that in the iteration, go and get another minimised value by one of the algorithms and use that in the second iteration, and so on.

i. Pseudo Code for implementation

- Equation: $X_{i+1} = X_i - \alpha^* \nabla f(x_i)$

Step 0: Select $X_0 \in R^n$, and i=0

Step 1: Compute $\nabla f(X_i)$

Step 2: if $\|\nabla f(X_i)\| < \mathcal{E}$, stop
otherwise go to step 3

Step 3: Update $\alpha^* = \text{argmin } f(X_i - \alpha\nabla f(X_i))$

Step 4: Compute $X_{i+1}=X_i - \alpha^*\nabla f(X_i)$

Step 5: Update i = i + 1

Step 6: Go to step 1

IV. Conclusion

- There are two different categories of optimization algorithms: those that use derivatives and those that don't.
- In conventional approaches, the first and sporadically second derivatives of the goal function are used.
- Direct search and stochastic methods are developed for objective functions in the absence of function derivatives.

3. Trials

i. Gradient descent method trials:

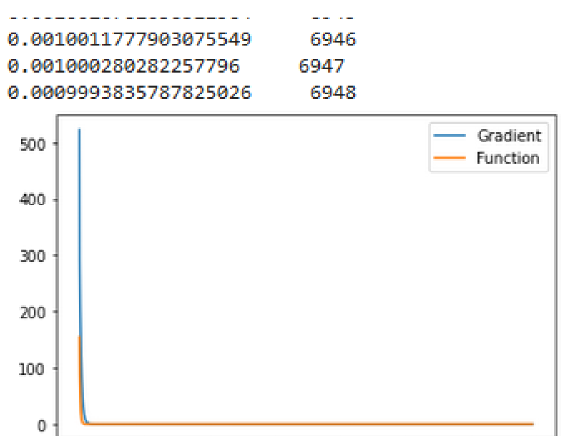

Figure 1. (0.43,0.48,0.3)

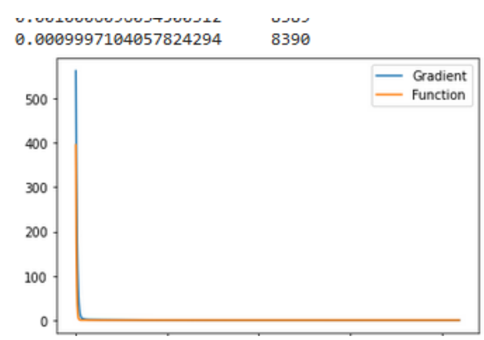

Figure 2. (0.3,0.8,0.93)

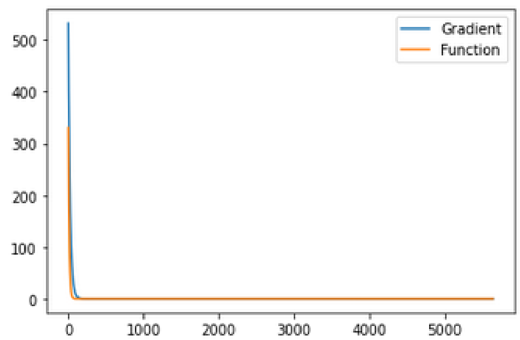

Figure 4. (0.5,0.6,0.8)

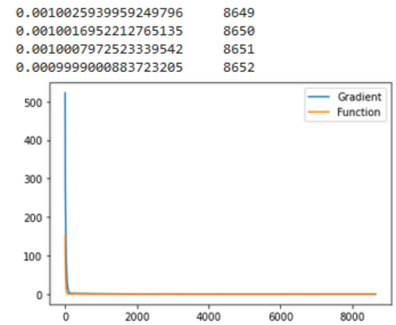

Figure 3. (0.743,0.348,0.3)

ii. Newton Raphson method trials:

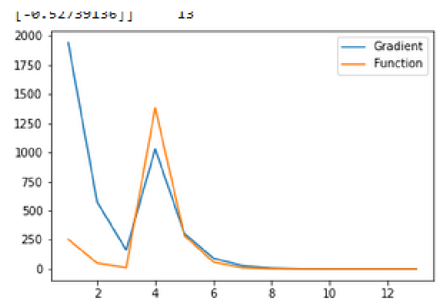

Figure 5. (0.3,0.8,0.93)

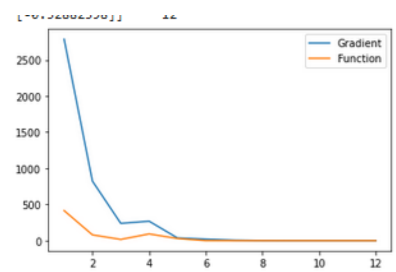

Figure 6. (0.4,0.8,0.9)

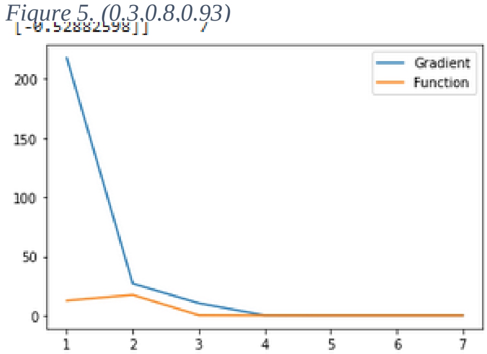

Figure 8. (0.7,0.3,0.3)

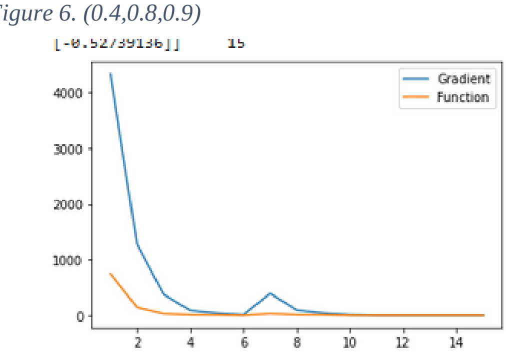

Figure 7. (0.567,0.942,0.834)

iii. Steepest descent method trials:

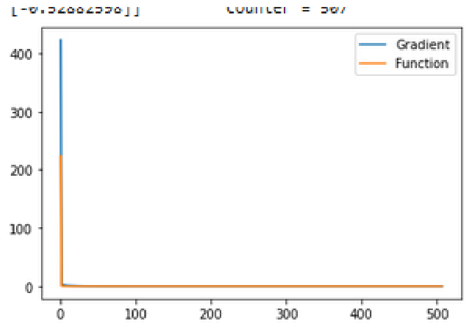

Figure 9. (0.8,0.9,0.9)

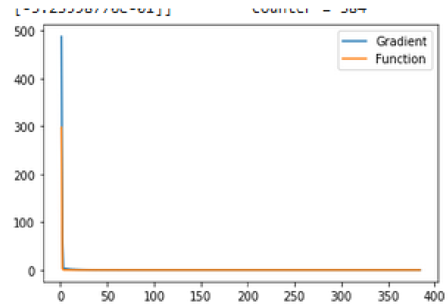

Figure 10. (0.67,0.982,0.721)

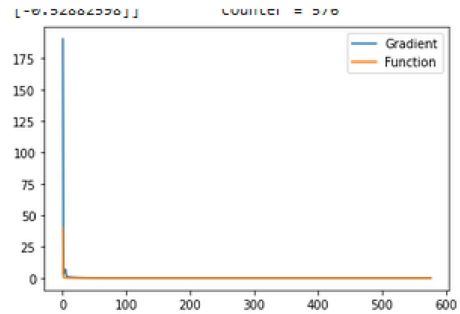

Figure 11. (0.677,0.268,0.185)

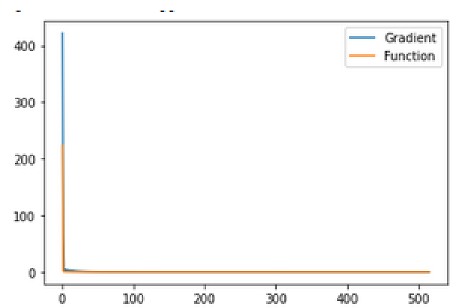

Figure 12. (0.768,0.866,0.559)

4. Appendix

Code of milestone 1

https://colab.research.google.com/drive/1Dy_Wm1hGT6JMdbdWp7g_ej-GIC8PZlOq?usp=sharing