Gradient Descent Algorithm Report

21900296 Yubin Park

21800090 Seonu Kim

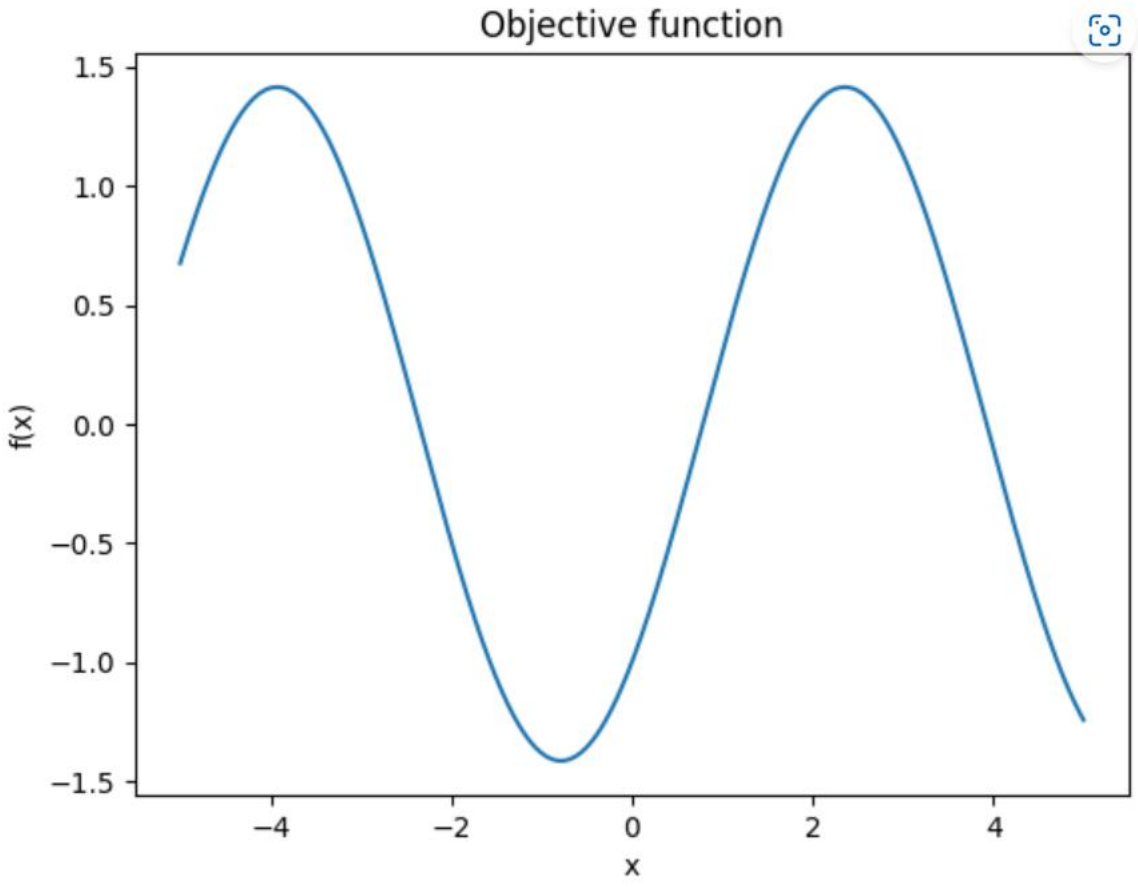
We are asked to find the minimum point of the objective function within $-5 \leq x \leq 5$:

$$f(x)=sin(x)-cos(x)$$

Deliverables for this problem are as follows:

- Create a plot showing the objective function within $-5 \leq x \leq 5$.
- Write a pseudo-code for your own algorithm.
- Explain how you design a stopping criterion.
- Discuss with your teammates and answer the following question:

1. Create a plot showing the objective function winthin $-5 \leq x \leq 5$



2. Write a pseudo-code for your own algorithm

1. Set Initial point X_i between $-5 \leq x \leq 5$
2. Set the step size(learning rate) α
3. set the steepest direction S_k
4. Set the max number of iterations N
5. Set the convergence threshold ϵ

Pseudocode

Input: objective function (cost function) $f(x)$,
Differentiated objective function $df(x)$,
initial point X_i,
Step Size α,
Steepest direction(partial derivative) S_k,
Max number of Iterations N,
Convergence threshold ϵ

Output: x values, y values of $f(x)$ at each point after every loop

set x = X_i
set x_values = [ ]
set y_values = [ ]

for $i = 1$ to N **do**
    $x_{new} \leftarrow x - \alpha S_k$
    if $x_{new} <$ -5 **then**
        x_{new} = -5
    if $x_{new} > 5$ **then**
        x_{new} = 5
    end if
    if $|f(x_{new} - f(x)| < \epsilon$ **then**
        exit for
    if $f(x_{new}) > f(x)$ **then**
        exit for
    end if
    x_values.append(x)
    y_values.append($f(x)$)
    x = x_{new}
end for
return x_values, y_values

3. Explain how you design a stopping criterion.

Stopping Criterion

A. If the Gradient Descent point gets close to the minimum, the step size will be very small, but we want to stop if it makes very small movements that is not significant. Therefore, we put convergence threshold to determine if we continue or not when the gradient has changed less than certain number.

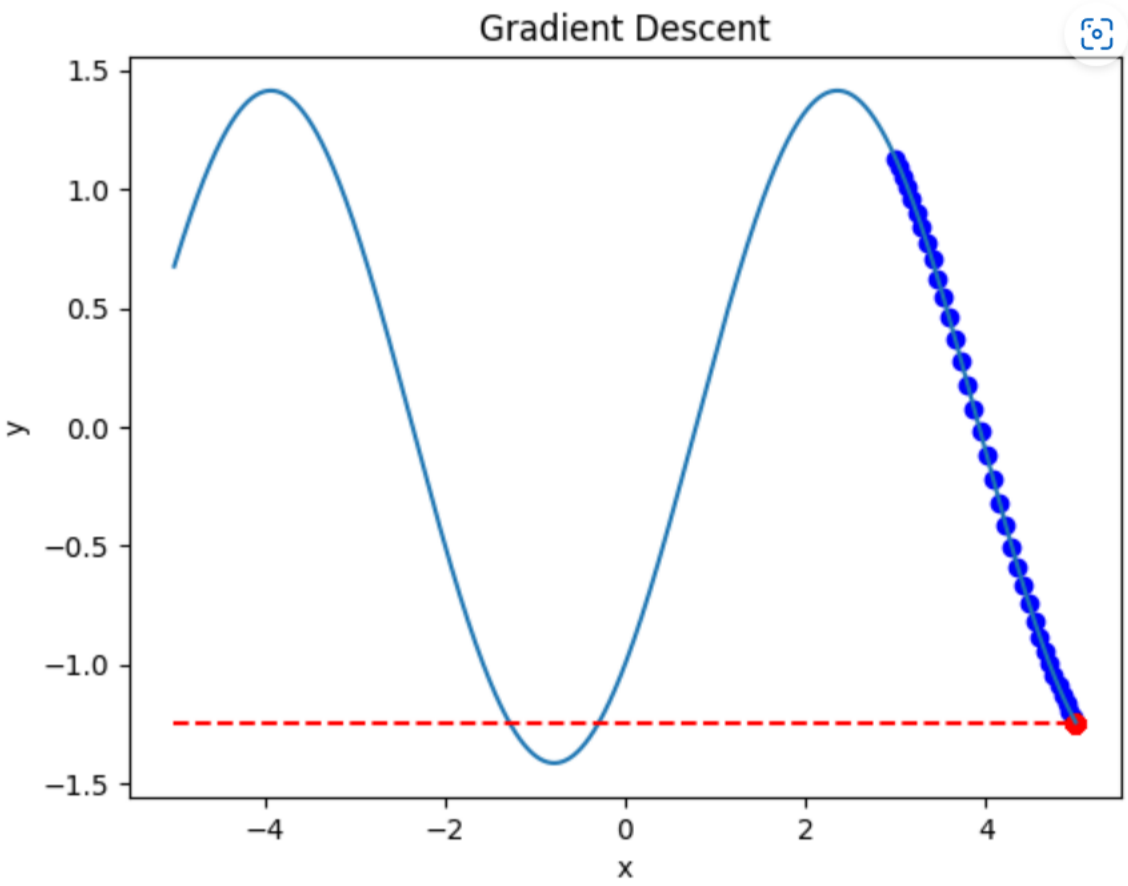
B. If the point of gradient descent gets close to the minimum, its y value will decrease. To stop the point at the minimum of the graph, we should stop the point movement before the f(x) value of updated x gets larger than that of x. Therefore, we put comparation of f(x) value of updated x and x to determine whether it continues or not.

4. Discuss with your teammates and answer the questions.

Q1. **Suppose that you implement a stopping criterion where the algorithm terminates when the number of iterations exceeds the maximum number of iterations. When the following conditions are given, what is the optimal solution? Do you think that it is a global optimum within the pre-determined range? If not, what is your strategy to improve the fidelity of the solution provided by the algorithm?**

- Initial point x = 3
- Step size = 0.05
- The maximum number of iterations = 500

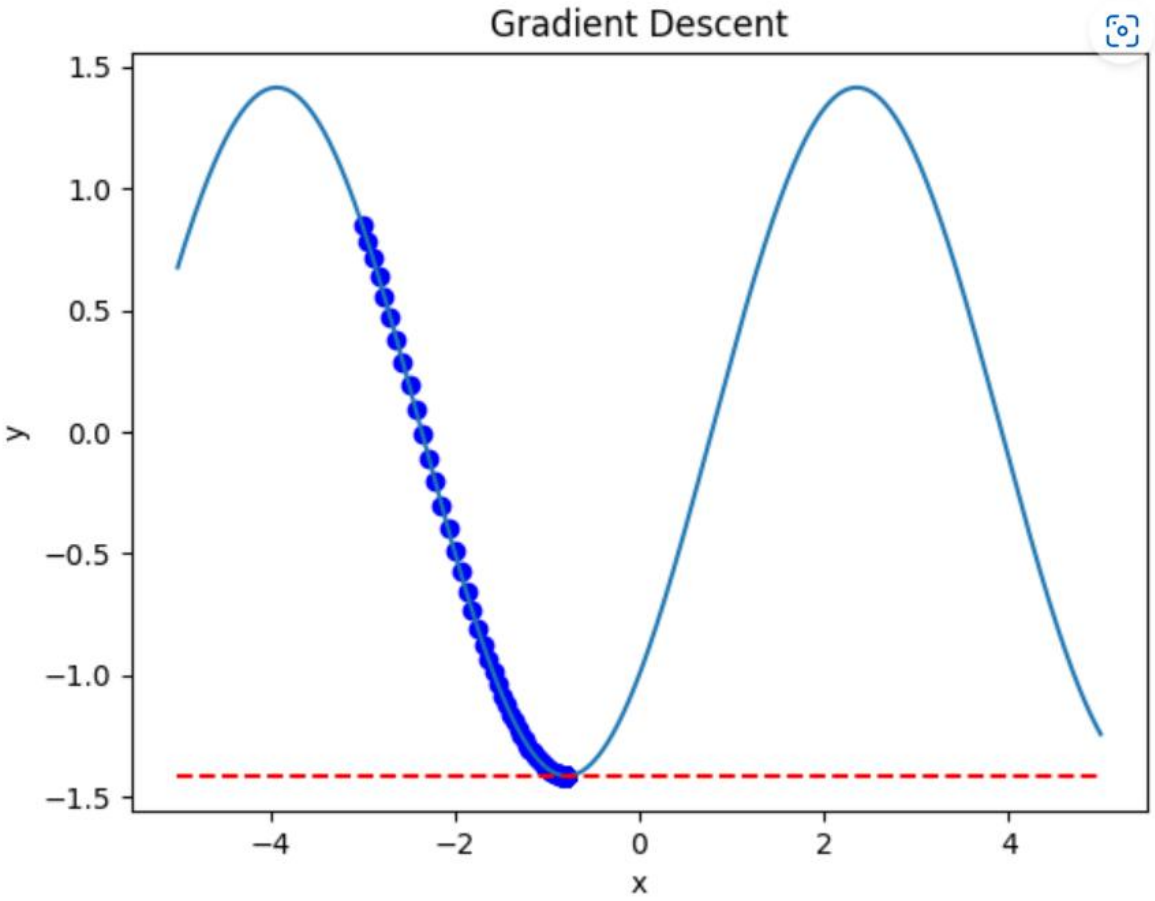
```
x= 5, y= -1.2425864601263648
```



The optimal solution was x= 5, y= -1.2425864601263648.
We think it is not global optimum within the pre-determined range as you can see in the graph. I think changing the initial point before the concave($-4 \le x \le -1$) can be feasible to find the global minimum. We anticipate that setting the initial point to ($-4 \le x \le -1$) will gradually shift the x value to the global optimal solution through the $\mathrm{x(new)} = \mathrm{x(old)} - \mathrm{a*df(x)}$ equation. On the other hand, if the step size and interaction condition values are adjusted, the slope value at the initial point does not change, so we expected that it can't find the global optimum.

Solution 1-1 : Change the initial point to in the concave part($-4 \leq x \leq -1$)

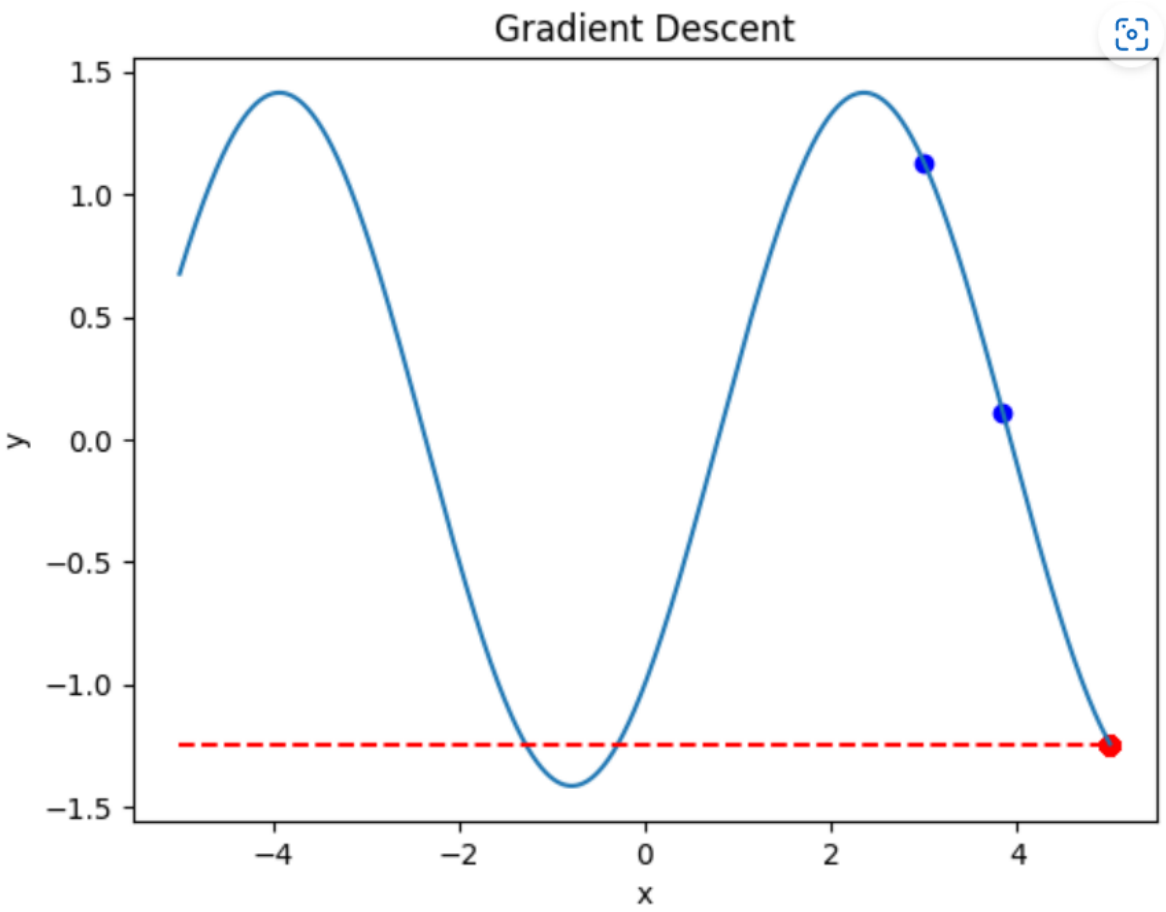
- Initial point x = - 3
- Step size = 0.05
- The maximum number of iterations = 500




```
Minimum: x= -0.785398166472164, y= -1.4142135623730951
Execution time: 0.0032320022583007812
```

Solution 1-2 : Increase the step size

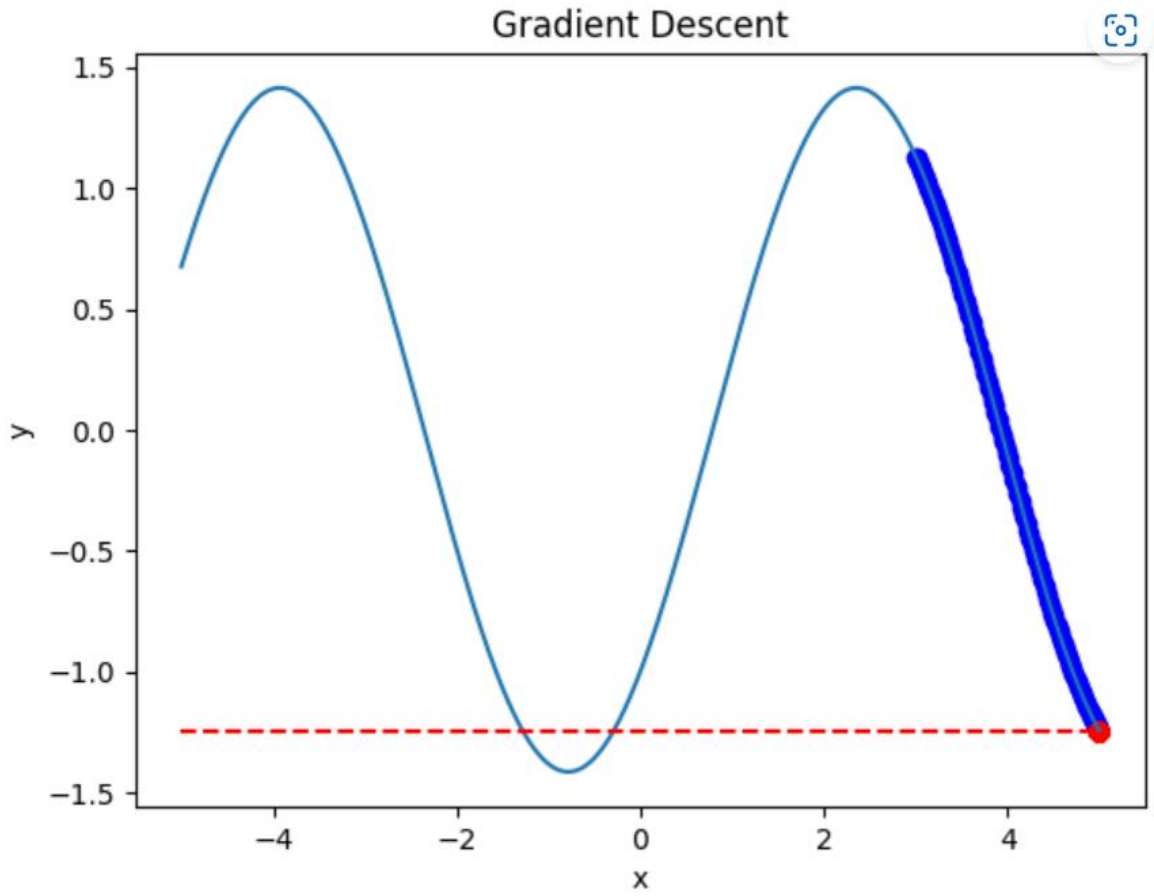
- Initial point x = 3
- Step size = 1
- The maximum number of iterations = 500




```
Minimum: x= 5, y= -1.2425864601263648
Execution time: 0.003902912139892578
```

Solution 1-3 : Increase the max number of iterations

- Initial point x = 3
- Step size = 1
- The maximum number of iterations = 500




```
Minimum: x= 5, y= -1.2425864601263648
Execution time: 0.018193483352661133
```

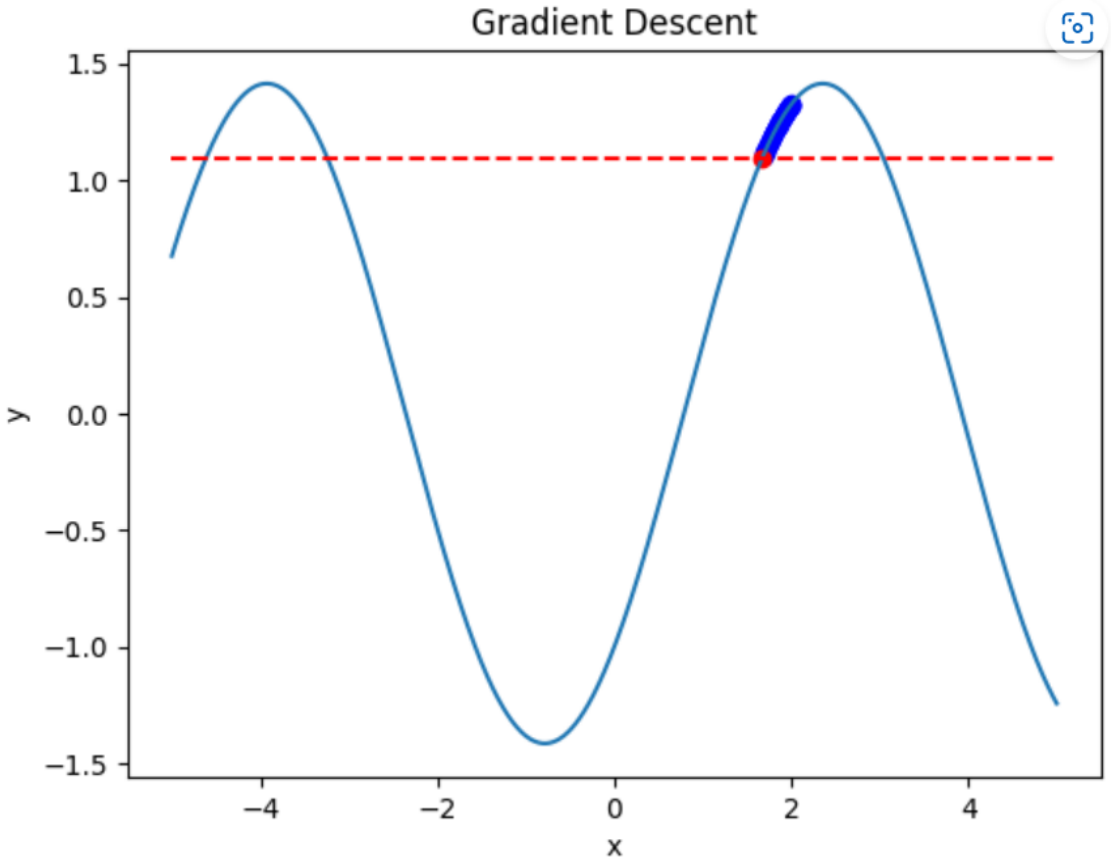
The solution 1-1 is the most feasible to find the global minimum.

	Parameters	Minimum	Execution Time (ms)	Did it find the minimum?
Initial	initial point = 3 max # of iterations: 500 Step Size = 0,05	x= 5 y= -1.2425864601263648	3.327608108520508	X
Solution 1-1	**initial point = -3** max # of iterations: 500 Step Size = 0,05	x= -0.785398166472164 y= -1.4142135623730951	3.2320022583007812	**O**
Solution 1-2	initial point = 3 max # of iterations: 500 **Step Size = 1**	x= 5 y= -1.2425864601263648	3.902912139892578	X
Solution 1-3	initial point = 3 **max # of iterations: 1000** Step Size = 0.05	x= 5 y= -1.2425864601263648	18.193483352661133	X

Looking at solutions 1-2 and 1-3, no meaningful results were obtained by increasing the values of step size and the max of iteration. However, when the initial point was set within the above range in Solution 1-1, the global optimal solution could be reached within the limited range. This result showed that our assumption was right.

Q2. **Suppose that you implement a stopping criterion where the algorithm terminates when the number of iterations exceeds the maximum number of iterations. When the following conditions are given, do you think that the algorithm provides a global optimum within the pre-determined range? If not, what is your strategy to improve the fidelity of the solution provided by the algorithm? You could set up a different initial point; however, it is assumed that the initial point cannot be changed. You are only allowed to control either step size or the maximum number of iterations. Which option would like to choose? What is the reason why you end up choosing the option?**

- Initial point x = 2
- Step size = 0.01
- The maximum number of iterations = 50

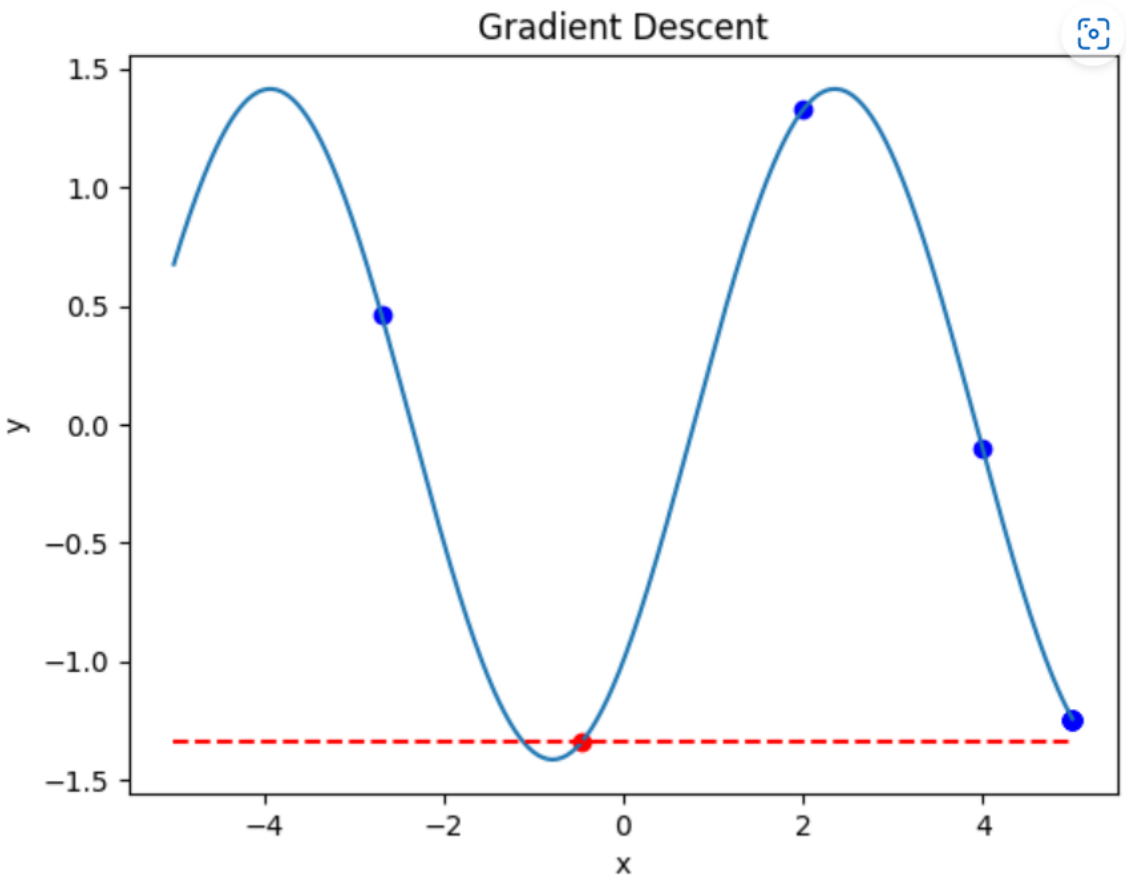



```
Minimum: x= 1.6678785359803567, y= 1.0922210041948084
Execution time: 0.00039505958557128906
```

We think this algorithm does not give a global optimum within the pre-determined range. In this case, increasing the step size and the max of iteration can be feasible if the initial point cannot be changed. If the maximum number of iterations gets larger, it will end up stopping at global minimum but execution time will increase. And we thought that if only the step size value is increased, the x value may reach the global optimal solution, but if the value is set too large, it may not be possible. So, we assume that the solution is to increase both the step size and the max of iteration appropriately.

Solution 2-1-1 : Increase the step size to 5

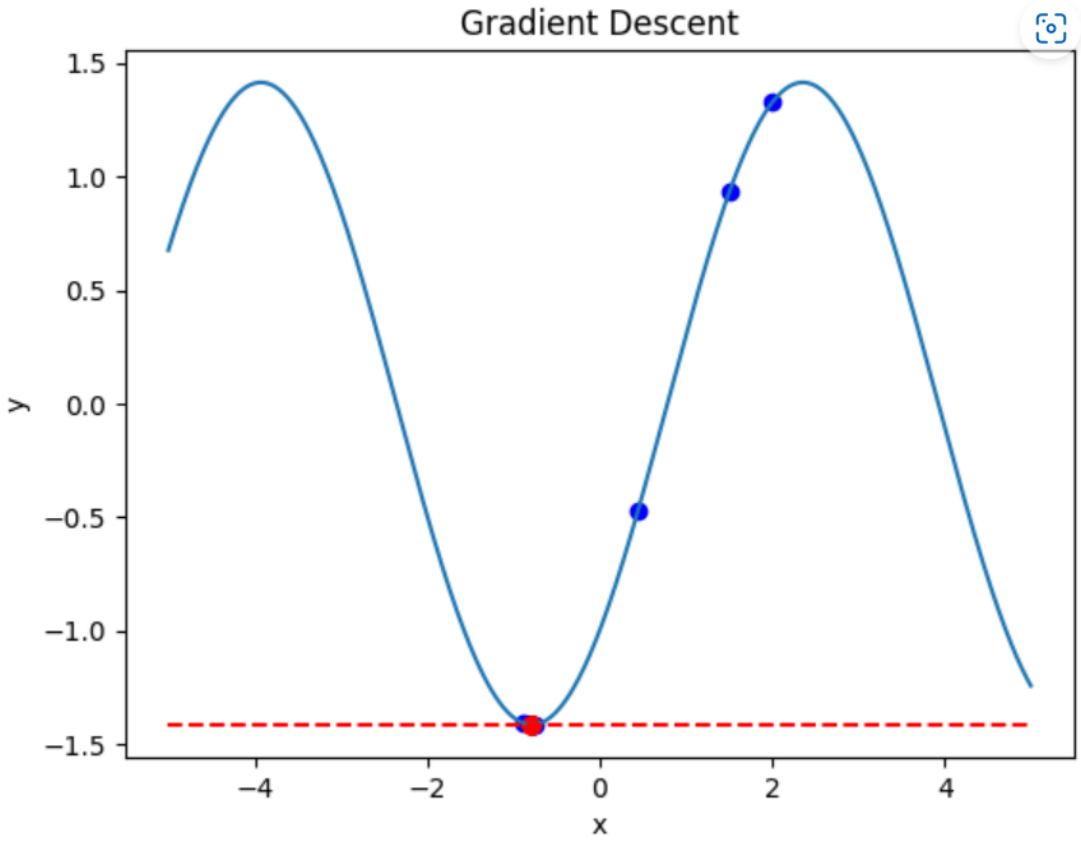
- Initial point x = 2
- Step size = 5
- The maximum number of iterations = 50




```
Minimum: x= -0.46575295139269635, y= -1.3425793501572731
Execution time: 0.0007793903350830078
```

Solution 2-1-2 : Increase the step size to 1

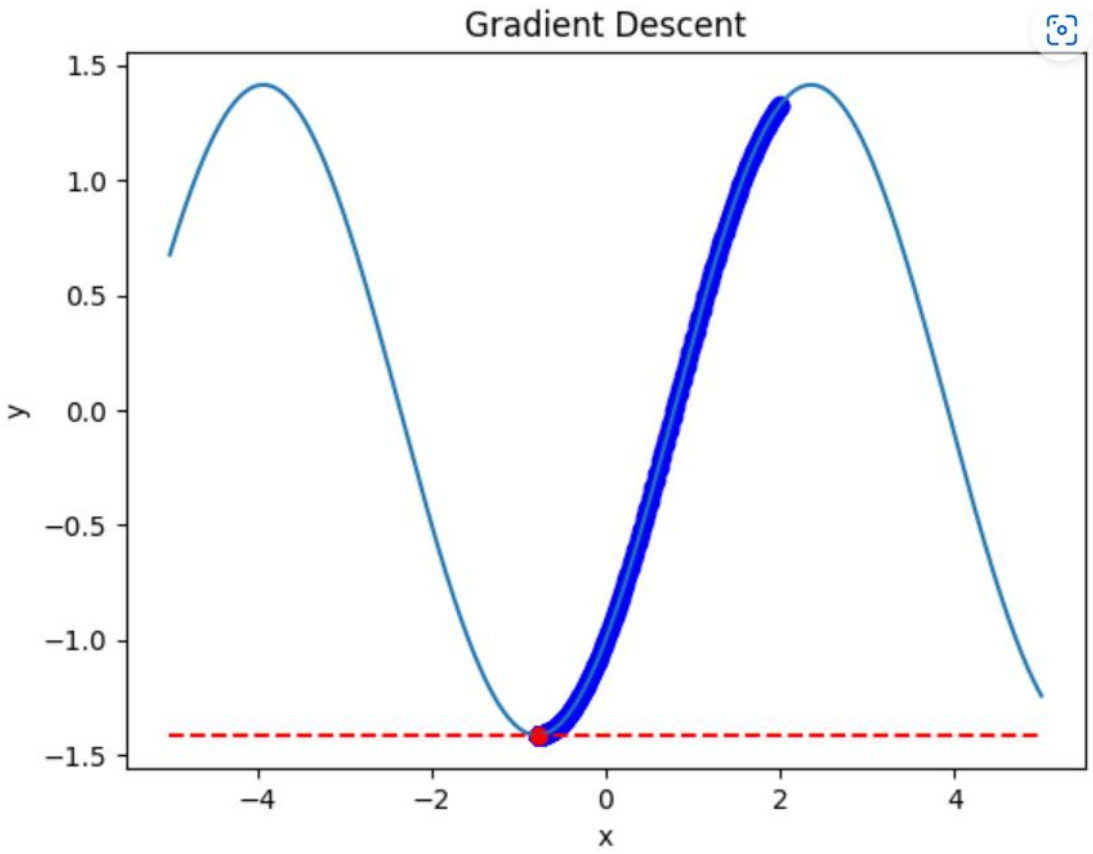
- Initial point x = 2
- Step size = 1
- The maximum number of iterations = 50




```
Minimum: x= -0.7853981765303392, y= -1.414213562373095
Execution time: 0.0003848075866699219
```

Solution 2-2 : Increase the max number of iteration

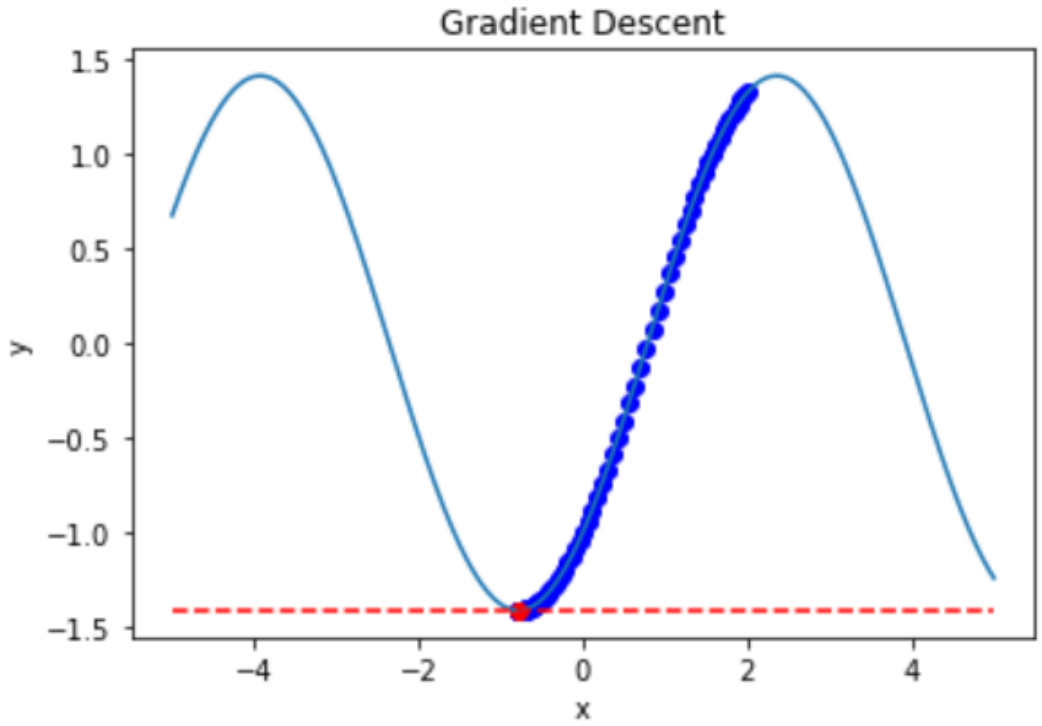
- Initial point x = 2
- Step size = 0.01
- The maximum number of iterations = 600




```
Minimum: x= -0.7831529918010731, y= -1.4142099979939138
Execution time: 0.004857540130615234
```

Solution 2-3 : Increase the max number of iteration and step size appropriately.

- Initial point x = 2
- Step size = 0.05
- The maximum number of iterations = 200




```
Minimum: x= -0.7853923603039141, y= -1.4142135623492824
Execution time: 0.0019993782043457703
```

Solution 2-3 is the most feasible solution to find the global minimum.

	Parameters	Minimum	Execution Time (ms)	Did it find the minimum?
Initial	initial point = 2 max # of iterations: 50 Step Size = 0,01	x= 1.6678785359803567 y= 1.0922210041948084	0.39505958557128906	X
Solution 2-1-1	initial point = 2 max # of iterations: 50 **Step Size = 5**	x= -0.4657529513926963 y= -1.3425793501572731	0.7793903350830078	X
Solution 2-1-2	initial point = 2 max # of iterations: 50 **Step Size = 1**	x= -0.7853981765303392 y= -1.414213562373095	0.3848075866699219	O
Solution 2-2	initial point = 2 **max # of iterations: 600** Step Size = 0.01	x= -0.7831529918010731 y= -1.4142099979939138	4.857540130615234	O
Solution 2-3	initial point = 2 **max # of iterations: 500** **Step Size = 0.05**	x= -0.7853981630347175 y= -1.4142135623730951	1.999378204345703	O

In solution 2-1-2, the global optimal was reached when the step size was increased. However, looking at solution 2-1-1, it was confirmed that if this value is setted too large, there is a possibility that the global optimal may not be reached. In solution 2-2, it can be seen that the global optimal can be reached when the step size is fixed and the max of arbitration value is increased. But you can see that the execution time takes a long time. Through this, it can be confirmed that it is not efficient to increase only the max number of iteration value. The solution 2-3 shows that when the step size and max number of iteration values are properly increased, the global optimal is found and the execution time is not large. Through this process, we can confirm that our assumption is correct.

Citations

1. [Pseudocode, Plotting a graph of an observation function, gradient descent algorithm](https://chat.openai.com/chat)

2. [Convergence threshold](https://stackoverflow.com/questions/17289082/gradient-descent-convergence-how-to-decide-convergence)

3. [Measurement of execution time] Lecture note(2-2 Performance Analysis) of 'Data structures in python' by professor Youngseop Kim

4. [Gradient Descent Algorithm equation] Lecture note(Module_4-1-1_Gradient_descent_algorithm) by professor Junghyun Kim

5. [Pseudocode](https://tex.stackexchange.com/questions/500546/algorithmicx-nested-for-loop-in-pseudocode)