Warsaw University of Technology
Faculty of Electronics and Information Technology
Specialization: Computer Science
Course: Introduction to Artificial Intelligence

PROJECT DOCUMENTATION
Simple Gradient Algorithm

Author:
Monika Jung
Student ID:
331384

This document describes the implementation of the gradient descent algorithm for function optimization. The algorithm minimizes two objective functions: a simple quadratic function and Matyas' function. The study involves analysing the impact of different learning rates (alpha values) on convergence and visualizing the optimization process for multiple starting points.

Implementation Details

The function gradient descent is an iterative optimisation algorithm, used to find a local minimum of a given function.

Parameters:

- f (callable): The objective function to minimize.
- start (tuple): The starting point (x1, x2).
- alpha (float): The learning rate.
- max_iter (int): Maximum number of iterations.
- tol (float): The tolerance for stopping based on gradient norm.

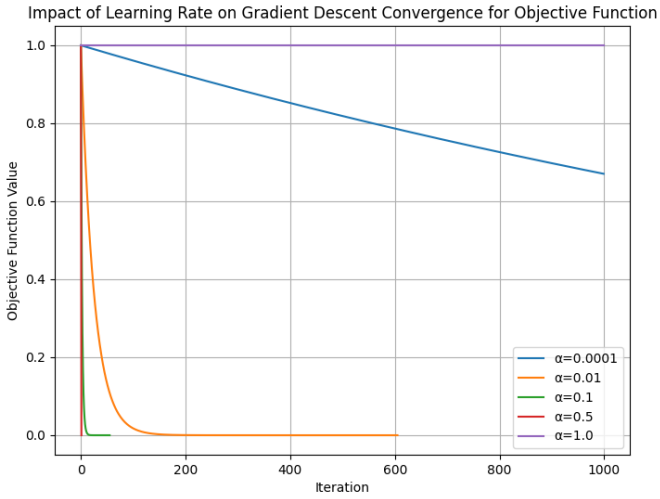
Returns:

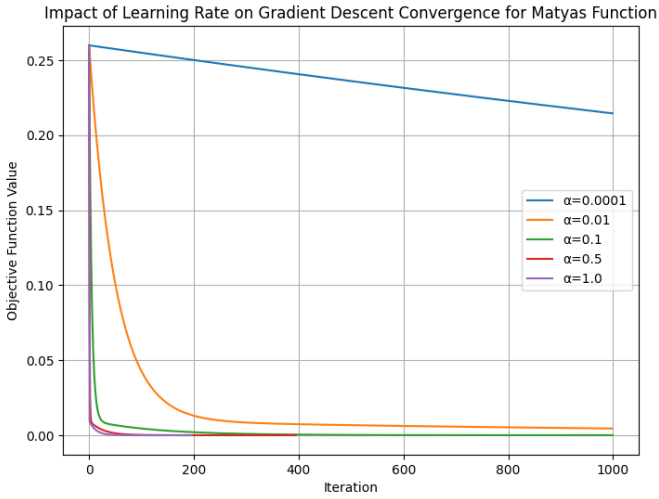
- trajectory (anp.array) The trajectory of points visited during the descent.
- function_values (list): The values of the objective function at each step.

Two functions are used:

1. f (x) = x1^2 + x2^2
2. Matyas function (for 2 dimensions)

Observations





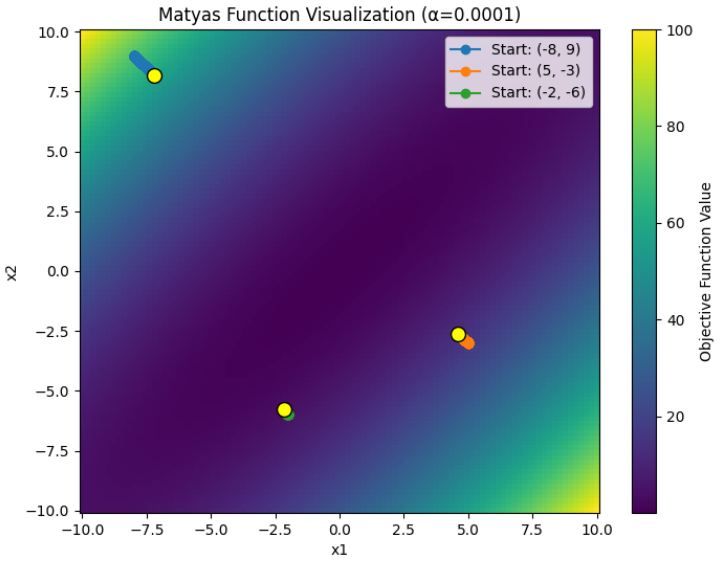
Different learning rates lead to the best results for two functions.

Visualization of Gradient Descent for Multiple Starting Points

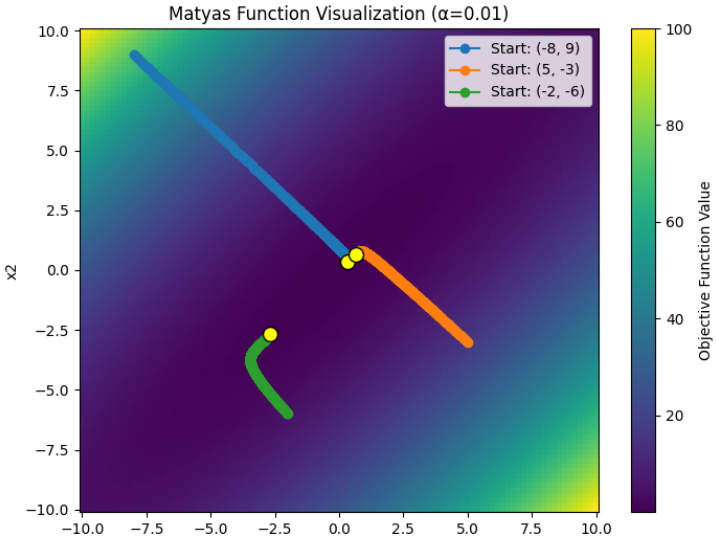
To analyse how the gradient descent algorithm behaves for different initial conditions, I tested three distinct starting points: **(-8, 9), (5, -3), (-2, -6),**

Matyas function plots:

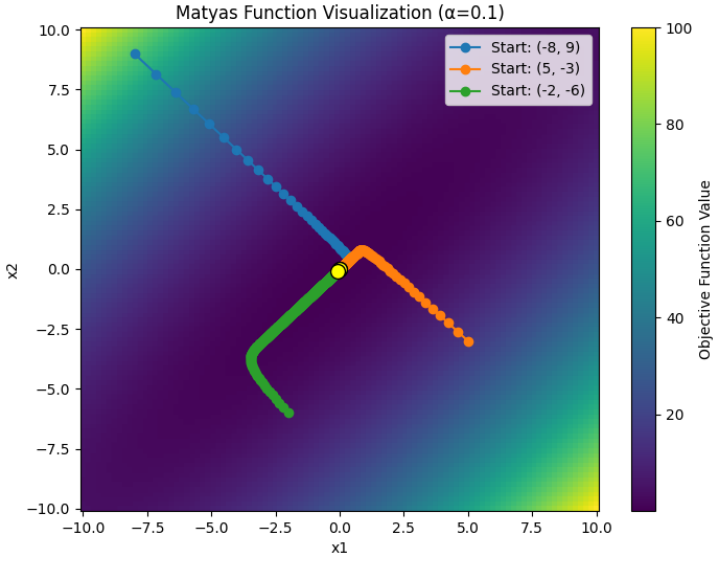
1) *Alpha = 0.0001*



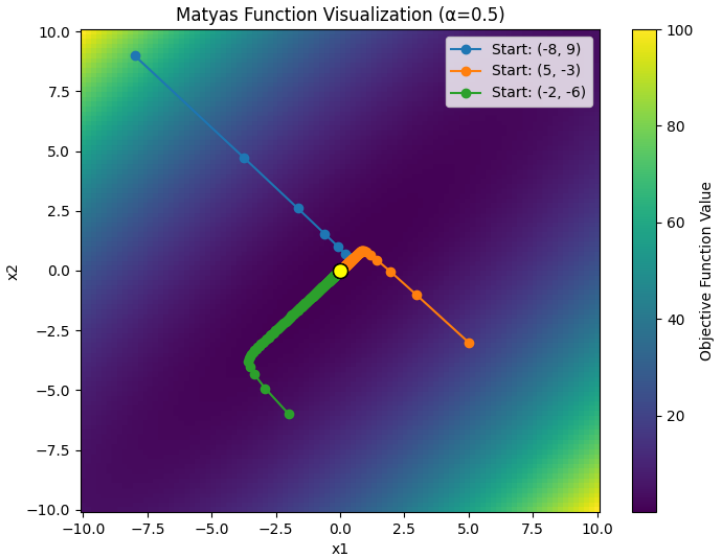
2) *Alpha = 0.01*



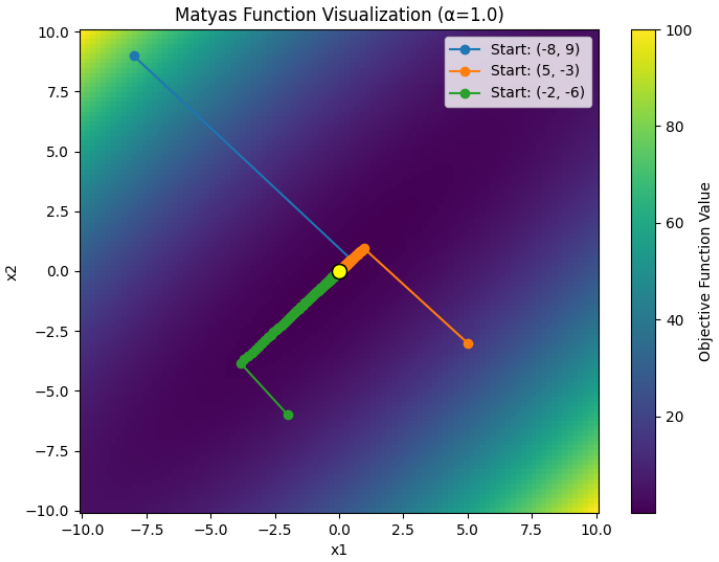
3) *Alpha = 0.1*



4) Alpha = 0.5



5) Alpha = 1.0



Matys function plot – conclusions :

The descent paths show curvature due to the interaction between x1 and x2 , indicating the influence of the -0.48x1x2 term.

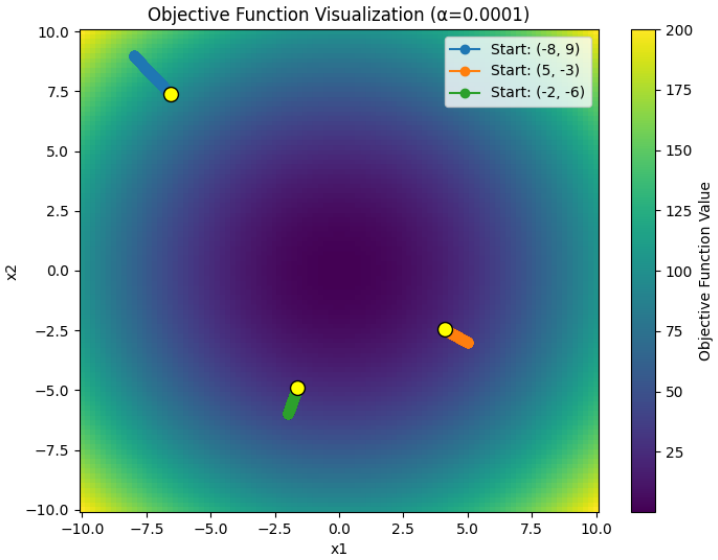
For α=0.0001, the function does not reach its minimum. This is likely due to the step size being too small, requiring significantly more iterations than the algorithm's limit allows.

For α=0.001, the function gets close to its minimum for two starting points. However, the algorithm stops when the gradient changes direction, which suggests that a higher number of iterations would be needed to achieve full convergence.

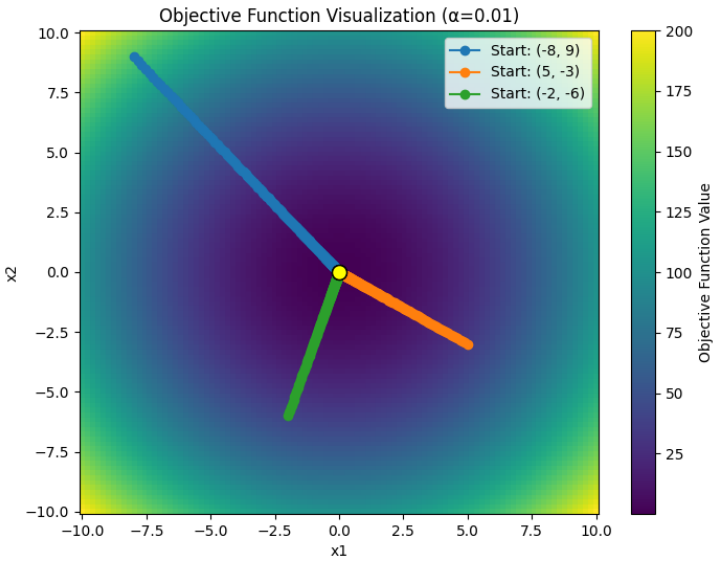
For higher values of α, the function successfully reaches its minimum. Additionally, we observe that the larger the learning rate, the fewer iterations are required to reach the minimum.

Objective function plots:

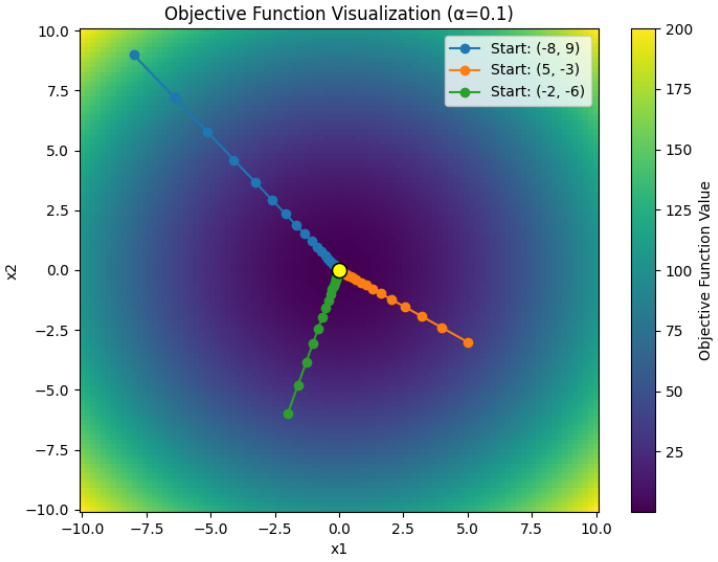
1) Alpha = 0.0001



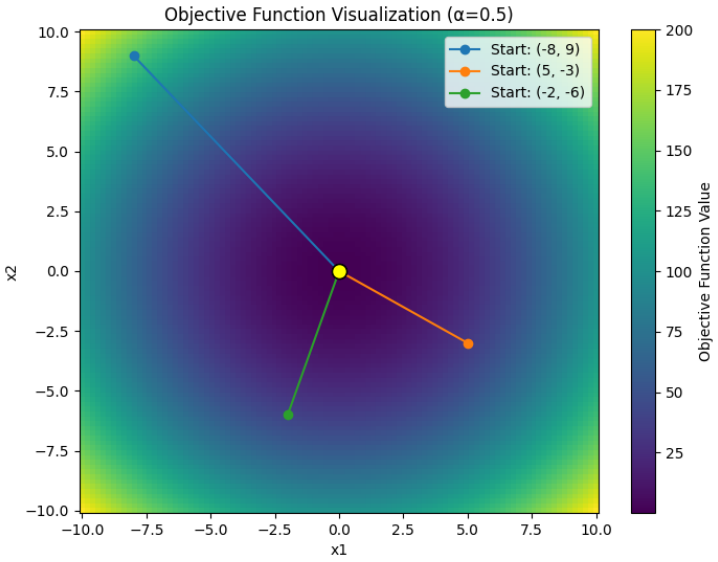
2) Alpha = 0.01



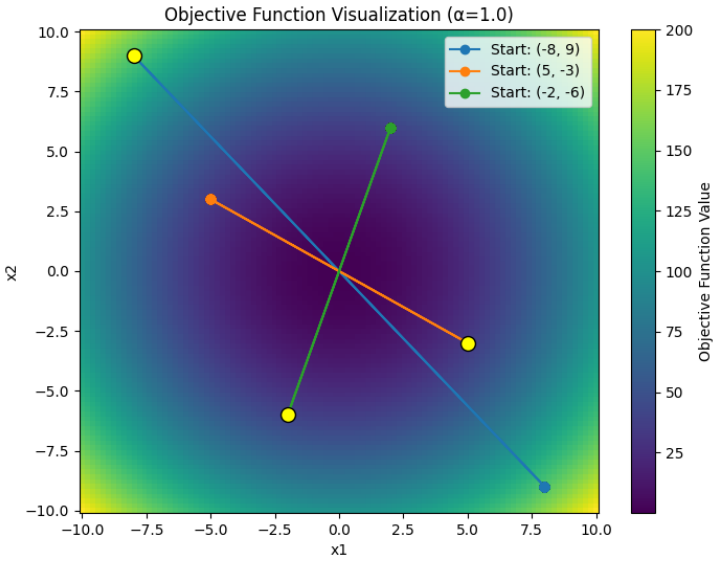
3) Alpha = 0.1



4) Alpha = 0.5



5) Alpha = 1.0



Objective function plot – conclusions:

For α=0.0001, the function does not reach its minimum. This is likely due to the step size being too small, requiring significantly more iterations than the algorithm's limit allows.

For α=0.001, α=0.1 and α=0.5, the function successfully reaches its minimum. Additionally, we observe that the larger the learning rate, the fewer iterations are required to reach the minimum.

For α=1.0 function does not reach its minimum. Instead, the function "jumps over" the minimum, indicating that the learning rate is too large