Report for Exercise No. 1

Adam Szostek, Index Number: 331443

February 11, 2025

1 Description of the Implemented Algorithm

The implemented algorithm is gradient descent. It is one of the optimization methods aimed at finding the minima of a function. The operation of the algorithm:

1. Initialization of the starting point $\mathbf{x}_0$.
2. Iteratively update the value of $\mathbf{x}$ according to the rule:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \beta \nabla f(\mathbf{x}_k)$$

where:

- β is the step size,
- $\nabla f(\mathbf{x}_k)$ is the gradient of the objective function at point $\mathbf{x}_k$.

3. Repeat iterations until the norm of the gradient reaches a value close to zero:

$$\|\nabla f(\mathbf{x}_k)\| \approx 0$$

The difference between consecutive points will be sufficiently small or until the maximum number of iterations (1500) is reached. The chosen precision is 1×10^{-6}.

2 Description of Planned Numerical Experiments

To test the convergence of the algorithm, three functions were used: a quadratic function and f3 and f12 from the CEC 2017 benchmark. For each function, the operation of the algorithm is presented for three values

of the step parameter β in graphs of the function values as a function of iteration steps. The initial point was randomly generated from the domain $[-100, 100]^{10}$. The graphs contain several curves representing the operation of the algorithm for different starting points. Some divergent graphs to ∞ have fewer iterations for readability.

3 Description of Obtained Results

3.1 Quadratic Function

For $\beta = 0.1$, the algorithm quickly finds the minimum of the function. For $\beta = 0.01$, we observe a tenfold increase in the algorithm's running time; however, the minimum is also found. For $\beta = 1.05$, the algorithm is divergent. The graphs show 3 trials for each parameter β.

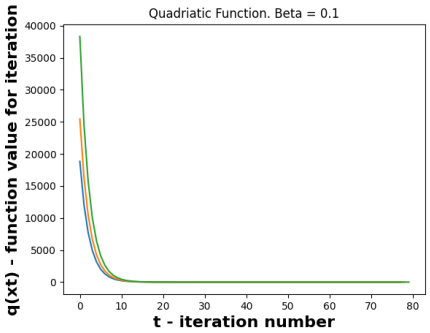

Figure 1: Convergence graph for the quadratic function, $\beta = 0.1$

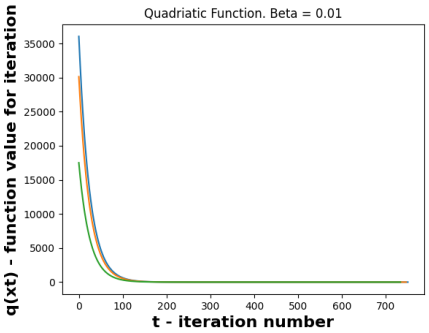

Figure 2: Convergence graph for the quadratic function, $\beta = 0.01$

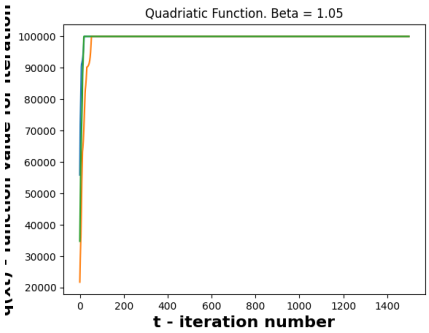

Figure 3: Convergence graph for the quadratic function, $\beta = 1.05$

	$\beta = 0.1$	$\beta = 0.01$	$\beta = 1.05$
Average Time	0.007 s	0.067 s	0.14 s

Table 1: Algorithm running time for the quadratic function.

3.2 Function f3

For f3, the algorithm reached the maximum number of iterations, thus not finding the minimum with the desired precision. Initially, the objective function values decrease rapidly, after which the algorithm reaches a point where further reduction in function values is minimal. It is worth noting that for the same β, depending on the starting point, the algorithm may be divergent or convergent. This indicates the complex structure of the CEC functions. The graphs use a logarithmic scale for the Y-axis, and each graph shows 3 trials for the parameter β (except the last one).

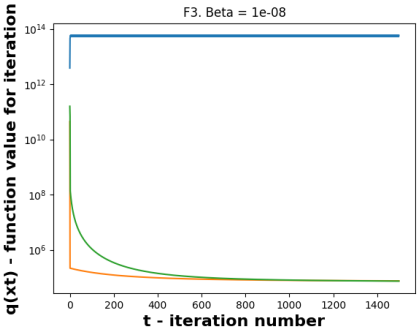

Figure 4: Convergence graph for f3, $\beta = 1 \times 10^{-8}$

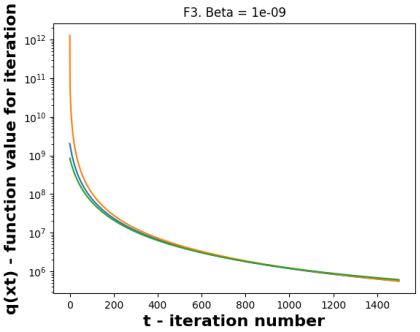

Figure 5: Convergence graph for f3, $\beta = 1 \times 10^{-9}$

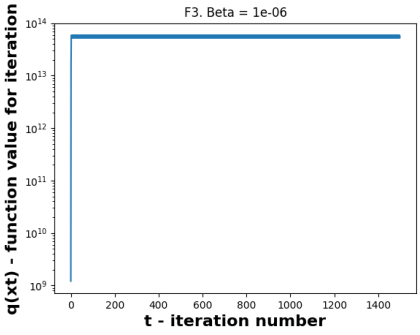

Figure 6: Convergence graph for f3, $\beta = 1 \times 10^{-6}$, 1 trial for readability.

	$\beta = 1 \times 10^{-9}$	$\beta = 1 \times 10^{-8}$	$\beta = 1 \times 10^{-6}$
Average Time	0.58 s	0.57 s	0.58 s

Table 2: Algorithm running time for function f3.

3.3 Function f12

The implemented algorithm did not find a minimum with the desired precision. For some β, we can observe that the function values converge up to a certain point, after which their change in further steps of the algorithm is negligible. Different function values at which the algorithm begins to slow down may indicate the existence of multiple local minima. For $\beta = 1 \times 10^{-6}$, we observe oscillation of the objective function values, which occurs when the parameter β is too large. In this case, the algorithm "jumps" around the minimum instead of approaching it stably. The graphs use a logarithmic scale for the Y-axis.

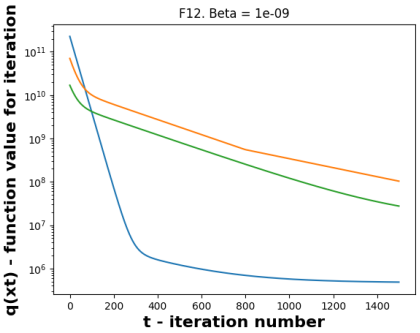

Figure 7: Convergence graph for f12, $\beta = 1 \times 10^{-9}$, 3 trials.

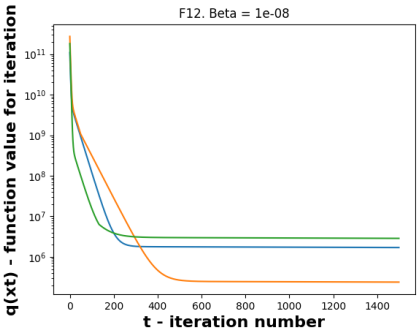

Figure 8: Convergence graph for f12, $\beta = 1 \times 10^{-8}$, 3 trials.

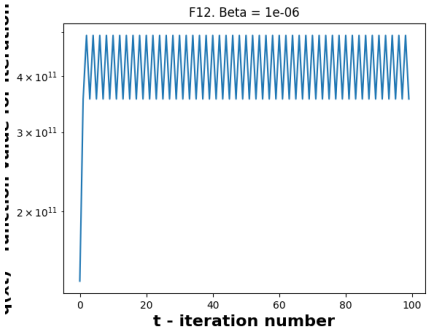

Figure 9: Convergence graph for f12, $\beta = 1 \times 10^{-6}$, 1 trial, fewer iterations for readability.

	$\beta = 1 \times 10^{-9}$	$\beta = 1 \times 10^{-8}$	$\beta = 1 \times 10^{-6}$
Average Time	1.51 s	1.50 s	1.53 s

Table 3: Algorithm running time for function f12.

4 Conclusions from the Conducted Studies

The gradient descent algorithm performs very well for simple functions such as the quadratic function. In the case of CEC2017 benchmark functions, which are much more complicated, the algorithm is not optimal. This occurs, among other reasons, because the function's gradient can easily get stuck on so-called plateaus, which are areas where the function is flat but not minima. This is clearly visible in the convergence graphs for f3 and f12,

where the algorithm, at a certain point, significantly slows down and ultimately does not find the minimum with the desired precision. Additionally, the gradient in "steep" areas of the function can have enormous values, which can lead to choosing a wrong search direction and, as a result, to the algorithm's divergence. It can also be stated that the algorithm is very sensitive to the step parameter β, as even the slightest changes can cause the algorithm to diverge. Importantly, the starting point is also crucial, as for the same β, the algorithm can be divergent or convergent depending on where it starts. For example, with the quadratic function, we observe that with a smaller step parameter β, there is a slower convergence to the minimum, whereas for larger β, it is faster.

In summary, the gradient descent algorithm is effective when we want to optimize a simple, uncomplicated function, but it requires careful selection of the step parameter β and the starting point.