Programming Assignment 1

Numerical Optimization

Ron Dagani 318170917

Github link

The project aims to implement and test line search minimization methods in Python, focusing on Gradient Descent and Newton methods with Wolfe condition and backtracking for step length search.

The project structure: two main directories: src and tests.
- src: contains modules for algorithm implementations and utility functions.
- tests: includes modules for testing and example functions.

Example functions are detailed in the next part of the report aside the result's plots for comparison: a. Contour lines of objective functions.
b. Function values at each iteration.

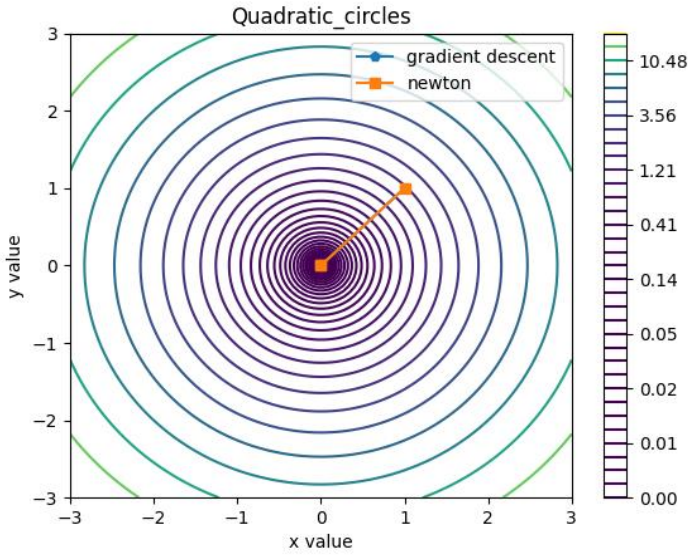
Further information regarding the implementation of the algorithms is available within the code documentation.

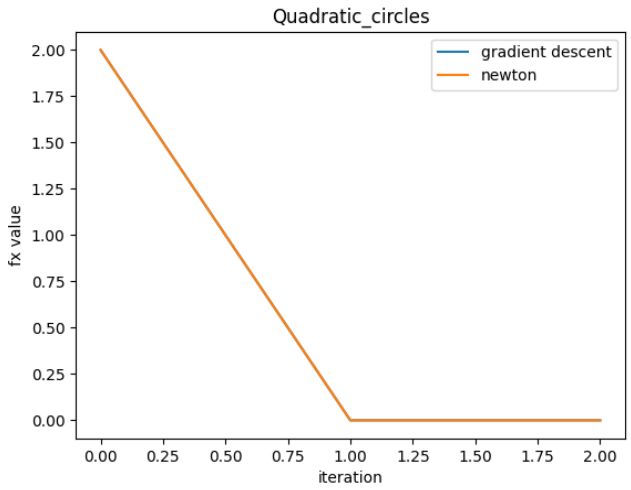
To conclude, Newton's method generally achieves rapid convergence across most examples. However, its dependency on the Hessian matrix may lead to computational hurdles and occasional failure in certain scenarios.
In contrast, Gradient Descent, being a first-order method, relies exclusively on the gradient. While it may exhibit slower convergence, occasionally requiring an impractical number of iterations, its reliance solely on the gradient proves advantageous in situations where calculating the Hessian poses challenges.

Results

1. Quadratic –

$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ (contour lines are circles)



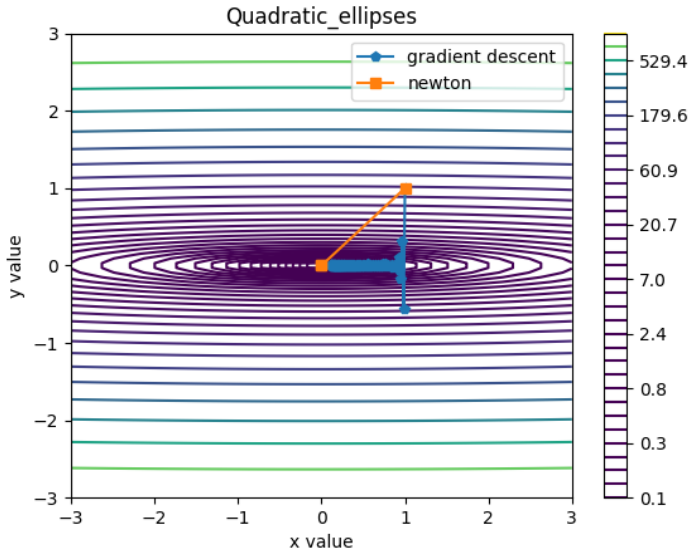


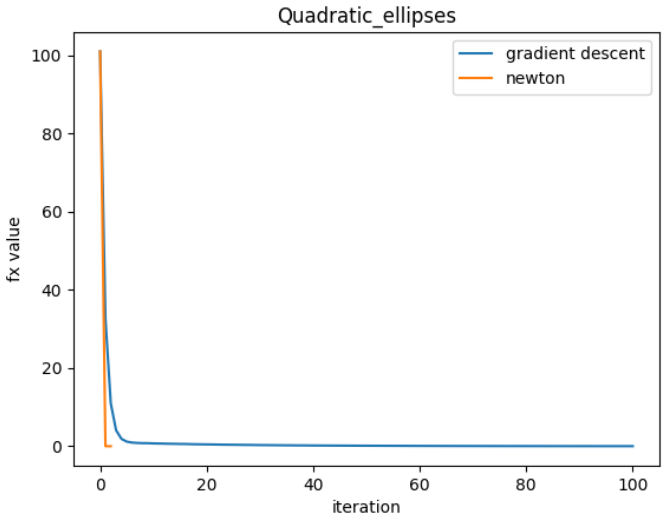

```
Iteration 1: f([0. 0.]) = 0.0
gradient stoped at iteration 2 successfully
```

```
Iteration 1: f([0. 0.]) = 0.0
Newton stoped at iteration 2 successfully
```

2. Quadratic –

$Q = \begin{bmatrix} 1 & 0 \\ 0 & 100 \end{bmatrix}$ (contour lines are axis aligned ellipses)



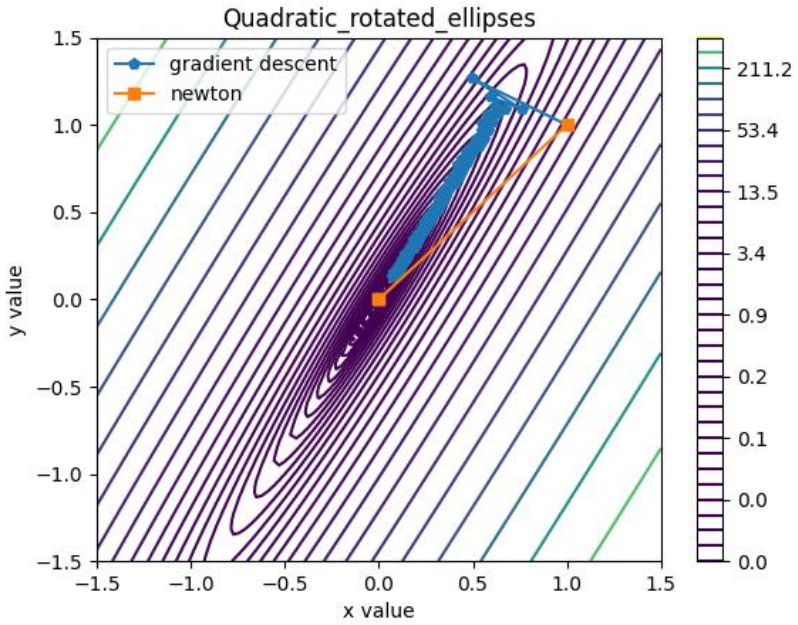


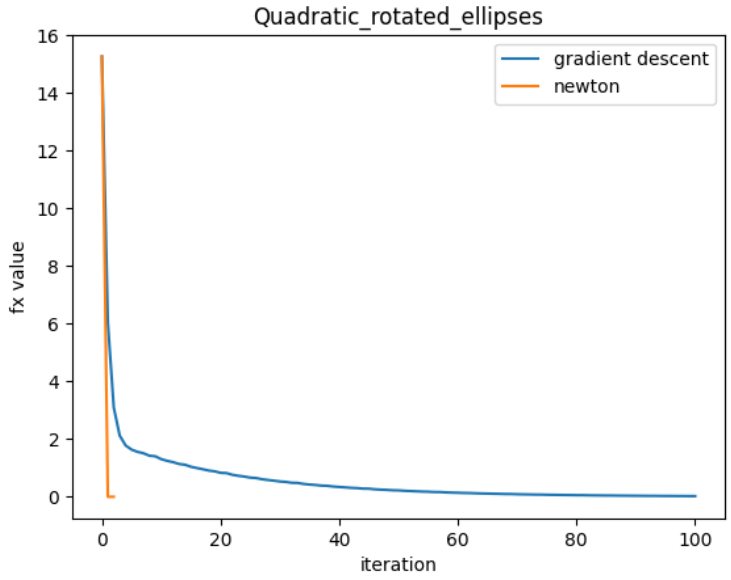

```
Iteration 99: f([ 0.11450918 -0.00157439]) = 0.013360223501345091
gradient stoped at iteration 100 Not successfully
```

```
Iteration 1: f([0. 0.]) = 0.0
Newton stoped at iteration 2 successfully
```

3. Quadratic –

$$Q = \begin{bmatrix} \frac{\sqrt{3}}{2} & -0.5 \\ 0.5 & \frac{\sqrt{3}}{2} \end{bmatrix}^T \begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{3}}{2} & -0.5 \\ 0.5 & \frac{\sqrt{3}}{2} \end{bmatrix}$$ (contour lines are rotated el

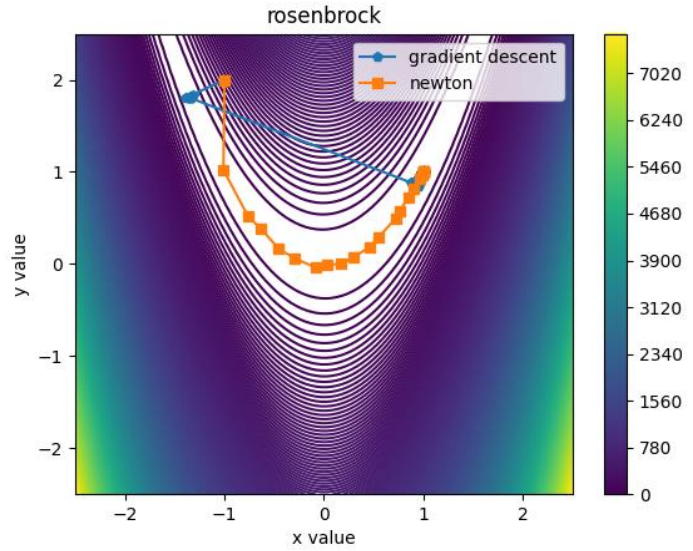


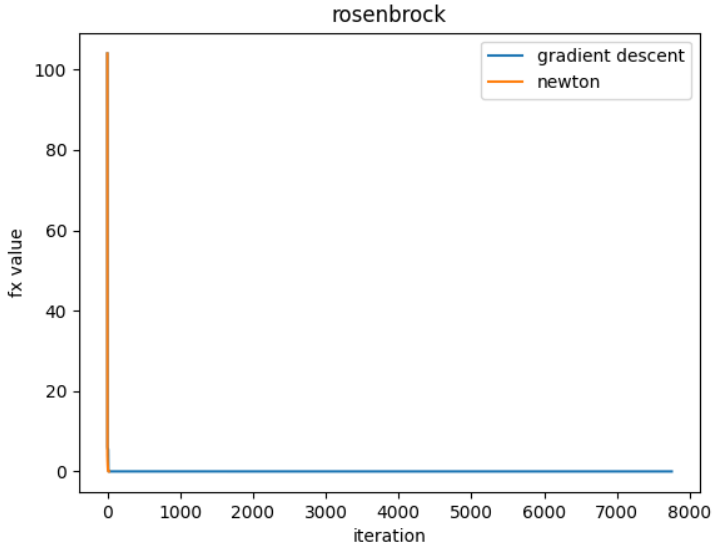



```
Iteration 99: f([0.07508442 0.13440407]) = 0.024171323786050 58
gradient stoped at iteration 100 Not successfully
```

```
Iteration 1: f([0. 0.]) = 0.0
Newton stoped at iteration 2 successfully
```

4. Rosenbrock function: $f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$

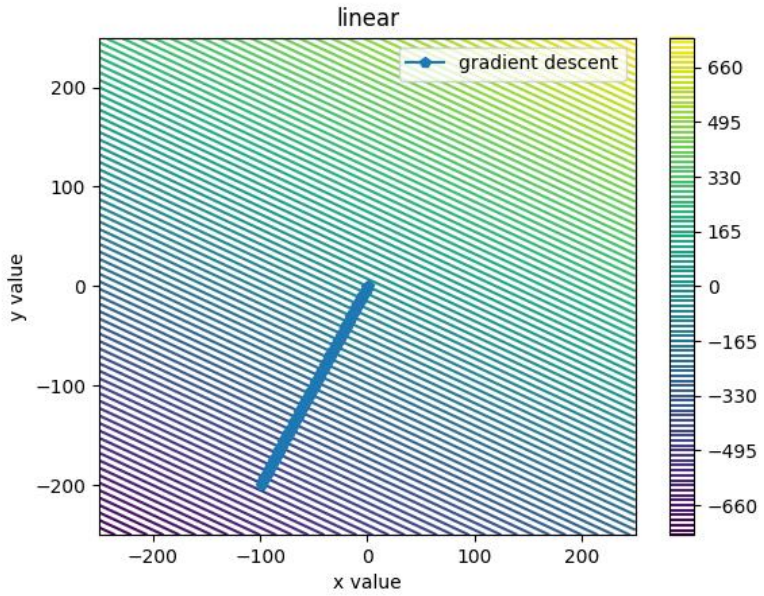


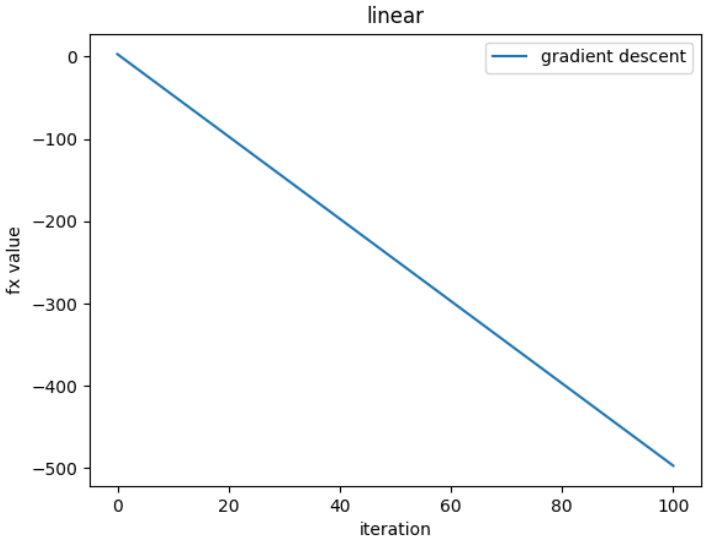



```
Iteration 7750: f([0.99989387 0.99978754]) = 1.1267921607826297e-08
gradient stoped at iteration 7751 successfully
```

```
Iteration 21: f([1. 1.]) = 6.668339839446365e-30
Newton stoped at iteration 22 successfully
```

5. linear function $f(x) = a^T x$

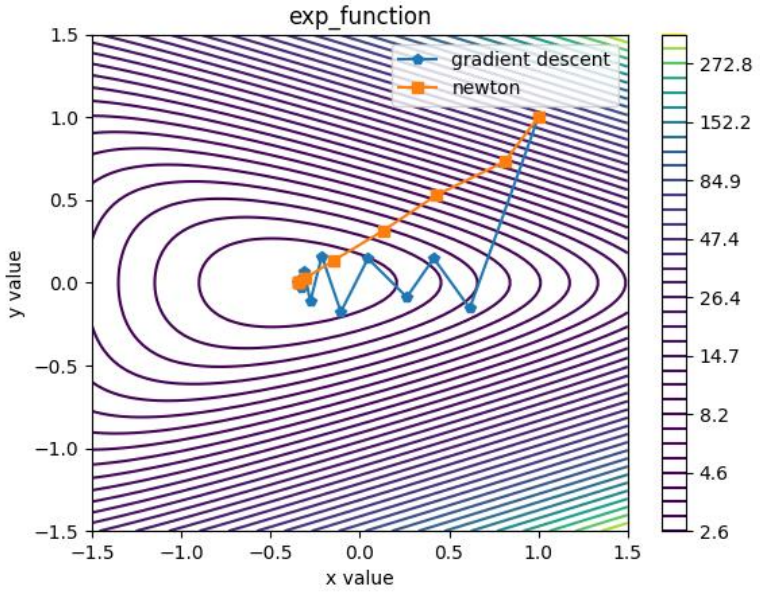


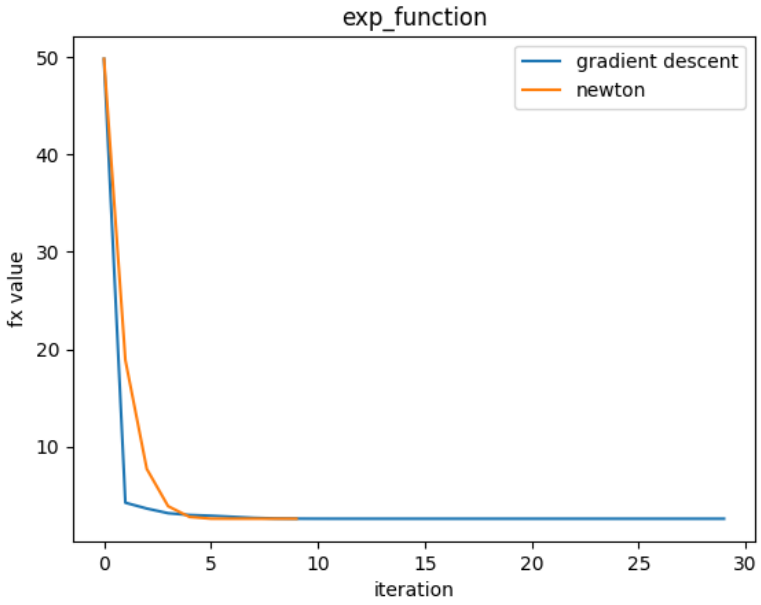



```
Iteration 99: f([ -98. -197.]) = -492.0
gradient stoped at iteration 100 Not successfully
```

For newton method, the hessian is not defined therefore it fails.

6. $f(x_1, x_2) = e^{x_1+3x_2-0.1} + e^{x_1-3x_2-0.1} + e^{-x_1-0.1}$






```
Iteration 28: f([-3.46571635e-01  5.19800828e-07]) = 2.5592666966646647
gradient stoped at iteration 29 successfully
```

```
Iteration 8: f([-3.46573590e-01  6.80690347e-12]) = 2.5592666966582156
Newton stoped at iteration 9 successfully
```