Lab 1: Function Minimization Using Newton's Method

Course: Introduction to Artificial Intelligence

Students:
Cagla Kuleasan 317287

Aayush Rautela 317746

Variant: 4

Instructor: Valeriya Khan

1. Introduction

In this lab, Newton's Method was implemented to minimize a two-variable mathematical function. This method is a second-order optimization algorithm that uses both gradient and Hessian information to find local minima of a given function. Unlike first-order methods like gradient descent, Newton's method leverages curvature information, often resulting in faster convergence near the solution.

The function we were tasked with minimizing is:

$$f(x, y) = 2\sin(x) + 3\cos(y)$$

This function is non-convex and has multiple local minima, making it a suitable candidate to analyze the behavior of Newton's method in different regions of the function domain.

2. Implementation Summary

The Newton's method algorithm was implemented in Python. The following key components were developed:

- **Gradient (∇f):** $\left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right] = [2\cos(x), -3\sin(y)]$
- **Hessian Matrix (Hf):** $\begin{bmatrix} -2\sin(x) & 0 \\ 0 & -3\cos(y) \end{bmatrix}$
- **Update Rule:** $x_{k+1} = x_k - \alpha H^{-1}(x_k)\nabla f(x_k)$

Where α is the step size.

A path-tracking mechanism was implemented to collect and visualize each step the algorithm took toward the minimum.

Visualization was done using matplotlib to generate 3D surface plots of the function, with the optimization path overlayed in red.

3. Test Cases and Results

Five test cases were executed using different initial guesses and learning rates. The goal was to observe how Newton's method behaves under varied conditions.

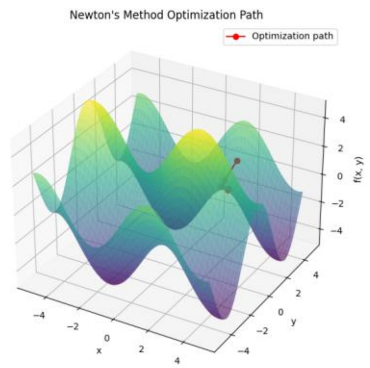
Test Case	**Initial Guess**	**Learning Rate**	**Iterations**	**Minimum Approximation (Rounded)**	**Notes**
1	[2.0, 2.0]	1.0	7	(1.5710, 3.1416)	Fast convergence to local minimum
2	[-3.0, 3.0]	0.5	21	(1.5708, 3.1416)	Slower, more cautious convergence
3	[0.0, 0.0]	1.0	1	(0.0, 0.0)	Stopped early (Hessian singular)
4	[-2.0, -2.0]	0.8	11	(-1.5710, -3.1416)	Converged to a negative minimum
5	[4.0, -4.0]	1.2	12	(4.7124, -3.1416)	Stable and fast convergence

4. Visualizations

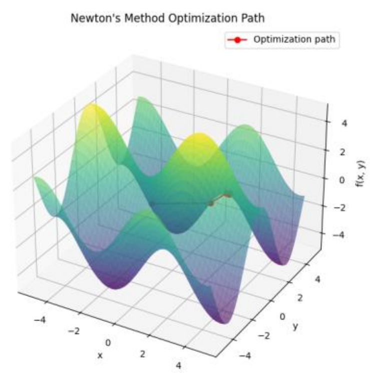
Each test case produced a 3D plot to visually interpret the optimization process. These plots revealed:

- Paths vary in length and direction based on initial guess and learning rate.
- Newton's method adjusts the path size according to local curvature.
- In Test Case 3, the method terminated early due to a singular Hessian matrix, showcasing a limitation of the approach.

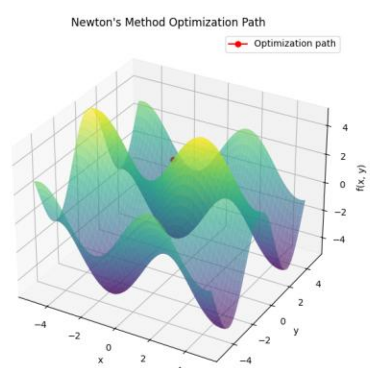
Included plots:



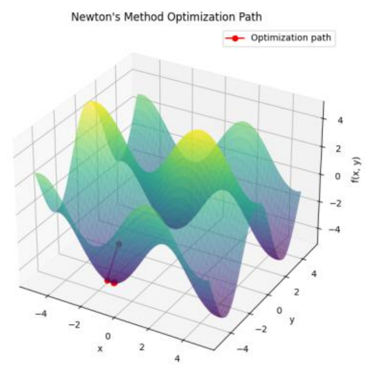
- plot_iterations_7.png



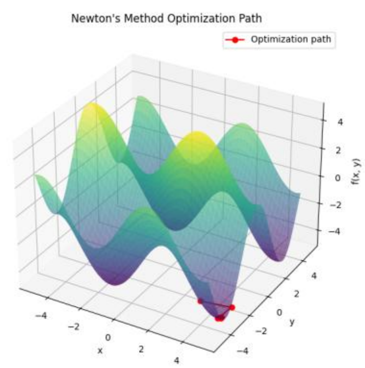
- plot_iterations_21.png



- plot_iterations_1.png



- plot_iterations_11.png



- plot_iterations_12.png

5. Conclusions

Newton's method successfully minimized the function in multiple regions. The test cases demonstrated its strengths:

- Fast convergence near well-behaved minima
- Accurate final values

However, the limitations were also clear:

- Requires a non-singular Hessian
- Sensitive to initial guess and step size
  Overall, this lab was a successful demonstration of second-order optimization in AI and highlighted both the power and pitfalls of Newton's method.