Optimization Algorithms: Steepest Descent & Newton's Method

This repository contains Python implementations of common unconstrained optimization algorithms: Steepest Descent and Newton's Method, along with several variants. The algorithms are tested on standard benchmark functions, and their performance is analyzed and visualized.

Features

- **Steepest Descent Algorithm** with various inexact line search strategies:
  - Backtracking with Armijo condition
  - Backtracking with Goldstein conditions
  - Bisection method with Wolfe conditions
- **Newton's Method** with modifications for robustness:
  - Pure Newton's Method (with basic regularization for singular Hessians)
  - Damped Newton's Method (using backtracking line search)
  - Levenberg-Marquardt modification
  - Combined Damped Newton + Levenberg-Marquardt approach
- **Benchmark Test Functions:**
  - Trid Function
  - Three Hump Camel Function
  - Rosenbrock Function
  - Styblinski-Tang Function
  - Simple Root of Squares Function (`func_1`)
- **Analytical Derivatives and Hessians:** Provided for all test functions.
- **Visualization:**
  - Convergence plots (Function Value vs. Iterations).
  - Gradient Norm plots (Gradient Norm vs. Iterations, log scale).
  - Contour plots with optimization path visualization for 2D functions.
- **Comprehensive Report:** Includes mathematical derivations, manual minima calculations, convergence analysis, and embedded plots (`boilerplate/report.md`).

PROF

Requirements

- Python 3.x
- NumPy
- Matplotlib
- PrettyTable

You can install the required libraries using pip:

```
pip install numpy matplotlib prettytable
```

Usage

To run all test cases using all implemented algorithms and generate the plots and summary table, execute the main script:

```
python3 main.py
```