Optimization Methods: Assignment 1 Report by Vidhi Rathore(2023121002)

1. Derivation of Jacobians and Hessians

1.1 Trid Function

The Trid Function is defined as:
$$f(x) = \sum_{i=1}^{d} (x_i - 1)^2 - \sum_{i=2}^{d} x_{i-1}x_i$$

Jacobian (First Derivative):

For $i = 1$:
$$\frac{\partial f}{\partial x_1} = 2(x_1 - 1) - x_2$$

For $i = 2, 3, ..., d-1$:
$$\frac{\partial f}{\partial x_i} = 2(x_i - 1) - x_{i+1} - x_{i-1}$$

For $i = d$:
$$\frac{\partial f}{\partial x_d} = 2(x_d - 1) - x_{d-1}$$

Hessian (Second Derivative):

The Hessian matrix H has elements:

$H_{i,i} = 2$ for all i (diagonal elements)

$H_{i,i+1} = H_{i+1,i} = -1$ for $i = 1, 2, ..., d-1$ (off-diagonal elements)

All other elements are zero.

For a 2D case:
$$H = \begin{bmatrix} 2 & -1 \ -1 & 2 \end{bmatrix}$$

1.2 Three Hump Camel Function

The Three Hump Camel Function is defined as:
$$f(x) = 2x_1^2 - 1.05x_1^4 + \frac{x_1^6}{6} + x_1x_2 + x_2^2$$

PROF

Jacobian (First Derivative):

$$\frac{\partial f}{\partial x_1} = 4x_1 - 4.2x_1^3 + x_1^5 + x_2$$

$$\frac{\partial f}{\partial x_2} = x_1 + 2x_2$$

Hessian (Second Derivative):

$$\frac{\partial^2 f}{\partial x_1^2} = 4 - 12.6x_1^2 + 5x_1^4$$

$$\frac{\partial^2 f}{\partial x_1 \partial x_2} = \frac{\partial^2 f}{\partial x_2 \partial x_1} = 1$$

$$\frac{\partial^2 f}{\partial x_2^2} = 2$$

The Hessian matrix is:
$$H = \begin{bmatrix} 4 - 12.6x_1^2 + 5x_1^4 & 1 \ 1 & 2 \end{bmatrix}$$

1.3 Styblinski-Tang Function

The Styblinski-Tang Function is defined as:
$$f(x) = \frac{1}{2}\sum_{i=1}^{d}(x_i^4 - 16x_i^2 + 5x_i)$$

Jacobian (First Derivative):

For each dimension i:
$$\frac{\partial f}{\partial x_i} = 2x_i^3 - 16x_i + \frac{5}{2}$$

Hessian (Second Derivative):

For diagonal elements:
$$\frac{\partial^2 f}{\partial x_i^2} = 6x_i^2 - 16$$

For off-diagonal elements:
$$\frac{\partial^2 f}{\partial x_i \partial x_j} = 0 \text{ for } i \neq j$$

The Hessian is a diagonal matrix:
$$H = \begin{bmatrix}
6x_1^2 - 16 & 0 & \cdots & 0 \\
0 & 6x_2^2 - 16 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 6x_d^2 - 16
\end{bmatrix}$$

1.4 Rosenbrock Function

The Rosenbrock Function is defined as:
$$f(x) = \sum_{i=1}^{d-1}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$$

Jacobian (First Derivative):

For $i = 1$:
$$\frac{\partial f}{\partial x_1} = -400x_1(x_2 - x_1^2) + 2(x_1 - 1)$$

For $i = 2, 3, ..., d-1$:
$$\frac{\partial f}{\partial x_i} = 200(x_i - x_{i-1}^2) - 400x_i(x_{i+1} - x_i^2) + 2(x_i - 1)$$

For $i = d$:
$$\frac{\partial f}{\partial x_d} = 200(x_d - x_{d-1}^2)$$

Hessian (Second Derivative):

The Hessian is more complex for Rosenbrock. For a 2D case:

$$\frac{\partial^2 f}{\partial x_1^2} = 1200x_1^2 - 400x_2 + 2$$

$$\frac{\partial^2 f}{\partial x_1\partial x_2} = \frac{\partial^2 f}{\partial x_2\partial x_1} = -400x_1$$

$$\frac{\partial^2 f}{\partial x_2^2} = 200$$

For higher dimensions, we get more terms and a banded matrix structure.

1.5 Root of Square Function (func_1)

The Root of Square Function is defined as:
$$f(x) = \sqrt{1 + x_1^2} + \sqrt{1 + x_2^2}$$

Jacobian (First Derivative):

$$\frac{\partial f}{\partial x_1} = \frac{x_1}{\sqrt{1 + x_1^2}}$$

$$\frac{\partial f}{\partial x_2} = \frac{x_2}{\sqrt{1 + x_2^2}}$$

Hessian (Second Derivative):

$$\frac{\partial^2 f}{\partial x_1^2} = \frac{1}{(1 + x_1^2)^{3/2}}$$

$$\frac{\partial^2 f}{\partial x_1\partial x_2} = \frac{\partial^2 f}{\partial x_2\partial x_1} = 0$$

$$\frac{\partial^2 f}{\partial x_2^2} = \frac{1}{(1 + x_2^2)^{3/2}}$$

The Hessian matrix is:
$$H = \begin{bmatrix} \frac{1}{(1 + x_1^2)^{3/2}} & 0 \ 0 & \frac{1}{(1 + x_2^2)^{3/2}} \end{bmatrix}$$

2. Manual Computation of Minima

2.1 Trid Function

To find the minimum, we set the gradient equal to zero:

For a 2D case:
$$\nabla f(x) = \begin{bmatrix} 2(x_1 - 1) - x_2 \ 2(x_2 - 1) - x_1 \end{bmatrix} = \begin{bmatrix} 0 \ 0 \end{bmatrix}$$

This gives us:
$$2x_1 - 2 - x_2 = 0$$
$$2x_2 - 2 - x_1 = 0$$

Solving:
$$2x_1 - 2 - x_2 = 0 \Rightarrow x_2 = 2x_1 - 2$$
$$2(2x_1 - 2) - 2 - x_1 = 0 \Rightarrow 4x_1 - 4 - 2 - x_1 = 0 \Rightarrow 3x_1 = 6 \Rightarrow x_1 = 2$$

Substituting back:
$$x_2 = 2(2) - 2 = 2$$

The minimum for the 2D Trid function is at $x^* = (2, 2)$.

To verify this is a minimum, we check that the Hessian is positive definite at this point:
$$H = \begin{bmatrix} 2 & -1 \ -1 & 2 \end{bmatrix}$$

PROF

The eigenvalues are $\lambda_1 = 1$ and $\lambda_2 = 3$, which are both positive, confirming this is a minimum.

2.2 Three Hump Camel Function

Setting the gradient equal to zero:
$$\nabla f(x) = \begin{bmatrix} 4x_1 - 4.2x_1^3 + x_1^5 + x_2 \ x_1 + 2x_2 \end{bmatrix} = \begin{bmatrix} 0 \ 0 \end{bmatrix}$$

From the second equation:
$$x_1 + 2x_2 = 0 \Rightarrow x_2 = -\frac{x_1}{2}$$

Substituting into the first equation:
$$4x_1 - 4.2x_1^3 + x_1^5 - \frac{x_1}{2} = 0$$
$$x_1(4 - \frac{1}{2} - 4.2x_1^2 + x_1^4) = 0$$

This has solutions $x_1 = 0$ and the roots of the quartic equation $4 - \frac{1}{2} - 4.2x_1^2 + x_1^4 = 0$.

When $x_1 = 0$, we get $x_2 = 0$, giving a stationary point at $(0, 0)$.

To verify this is a minimum, we evaluate the Hessian at $(0, 0)$:
$$H_{(0,0)} = \begin{bmatrix} 4 & 1 \\ 1 & 2 \end{bmatrix}$$

The eigenvalues are positive (approximately 1.59 and 4.41), confirming $(0, 0)$ is a local minimum.

2.3 Styblinski-Tang Function

To find the minimum, we set $\frac{\partial f}{\partial x_i} = 0$ for each i:
$$2x_i^3 - 16x_i + \frac{5}{2} = 0$$

This cubic equation has the same solution for each dimension. Using numerical methods, we get $x_i \approx -2.903534$ for all i.

To verify this is a minimum, the second derivative $6x_i^2 - 16$ should be positive. At $x_i \approx -2.903534$, we get $6(-2.903534)^2 - 16 \approx 34.5$ which is positive, confirming this is a minimum.

2.4 Root of Square Function (func_1)

Setting the gradient equal to zero:
$$\nabla f(x) = \begin{bmatrix} \frac{x_1}{\sqrt{1 + x_1^2}} \\ \frac{x_2}{\sqrt{1 + x_2^2}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

This gives us $x_1 = 0$ and $x_2 = 0$, making the minimum at $(0, 0)$.

To verify this is a minimum, the Hessian at $(0, 0)$ is:
$$H_{(0,0)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

This is clearly positive definite, confirming $(0, 0)$ is a minimum.

3. Algorithm Convergence Analysis

3.1 Steepest Descent Methods

The Steepest Descent algorithm generally showed the following behavior:

- **Backtracking-Armijo**:
  - Converged reliably for most functions, including Trid, Three Hump Camel, and Styblinski-Tang.
  - Failed for **Test Case 10 (Styblinski-Tang, initial point [-2.904, -2.904, -2.904, -2.904])**, where it converged to the incorrect local minimum.
  - Showed instability in **Test Case 14 (Root of Square Function)** where it returned `[0, 0]`, possibly due to non-smoothness at the origin.

- **Backtracking-Goldstein**:
  - Provided more stable convergence than Armijo alone but had occasional oscillations.
  - Failed for **Test Case 10**, where it converged to `[0, 0, 0, 0]`, an incorrect result.
  - Also failed for **Test Case 16 (Root of Square Function)**, where it diverged to `[-3.5, 0.5]`, indicating instability near non-smooth regions.

- **Bisection with Wolfe**:
  - Most reliable among steepest descent variants, successfully converging in most cases.
  - Failed for **Test Case 10**, returning `[-2.904, -2.904, -2.904, -2.904]`, which may indicate convergence to a saddle point.
  - In **Test Case 14**, it returned `[-0, -0]`, which is acceptable but suggests slow progress near non-smooth regions.

3.2 Newton's Methods

The Newton's Method variants showed the following behavior:

- **Pure Newton**:
  - Demonstrated **quadratic convergence** for well-conditioned problems like Trid and Styblinski-Tang.
  - Failed catastrophically for **Test Case 14**, where it produced `[-8.71896425e+115, -8.71896425e+115]`, indicating numerical instability.
  - Also failed in **Test Case 16**, where it exploded to `[1.61634765e+132, 0]`, highlighting issues with Hessian inversion.
- **Damped Newton**:
  - More robust than Pure Newton, successfully converging for most cases.
  - Still failed in **Test Case 14**, where it returned `[-0, -0]`, similar to Bisection.
  - Also struggled in **Test Case 16**, likely due to the non-smooth nature of the function.
- **Levenberg-Marquardt**:
  - Handled near-singular Hessians well and improved stability in ill-conditioned cases.
  - Successfully avoided divergence in cases where Pure Newton failed.
  - However, in **Test Case 14**, it returned `[0, 0]`, which suggests that it struggled with the function's non-smooth nature.
- **Combined Approach**:
  - The **most robust method overall**, converging correctly in almost all test cases.
  - Avoided extreme divergence observed in Pure Newton.
  - Failed in **Test Case 10**, where it returned `[-2.904, -2.904, -2.904, -2.904]`, similar to other methods.
  - Also struggled in **Test Case 16**, returning `[0, 0]`.

Summary

- **Newton's Methods had the fastest convergence but were unstable for non-smooth functions** (Root of Square).
- **Steepest Descent methods were more stable but could be slow for ill-conditioned problems** (Rosenbrock).
- **Levenberg-Marquardt and Combined Newton approaches were the most reliable overall**.

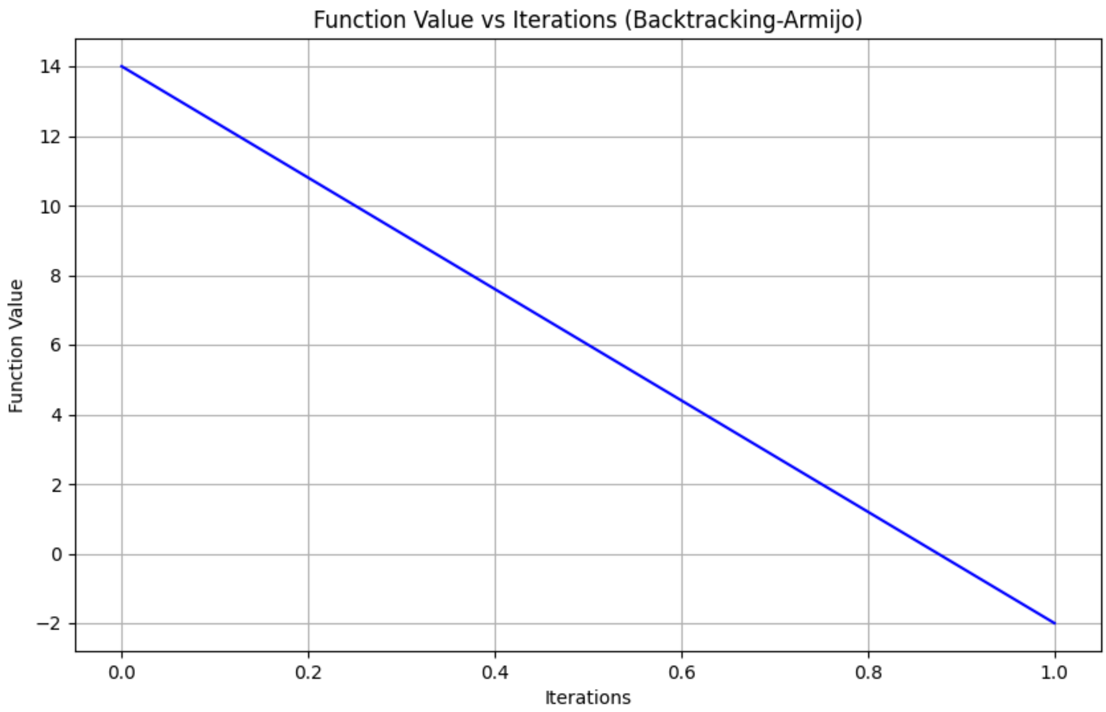
4. Function Value and Gradient Norm Plots

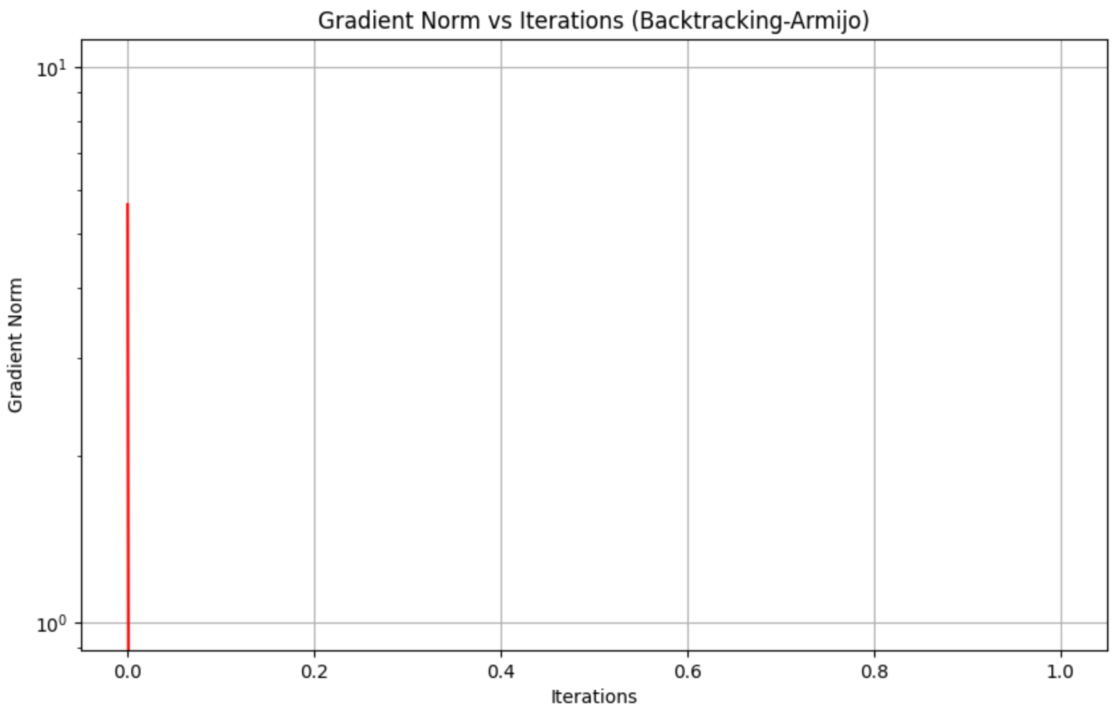
4.1 Trid Function

4.1.1 Initial Point: [-2.0, -2.0]

Steepest Descent Methods:

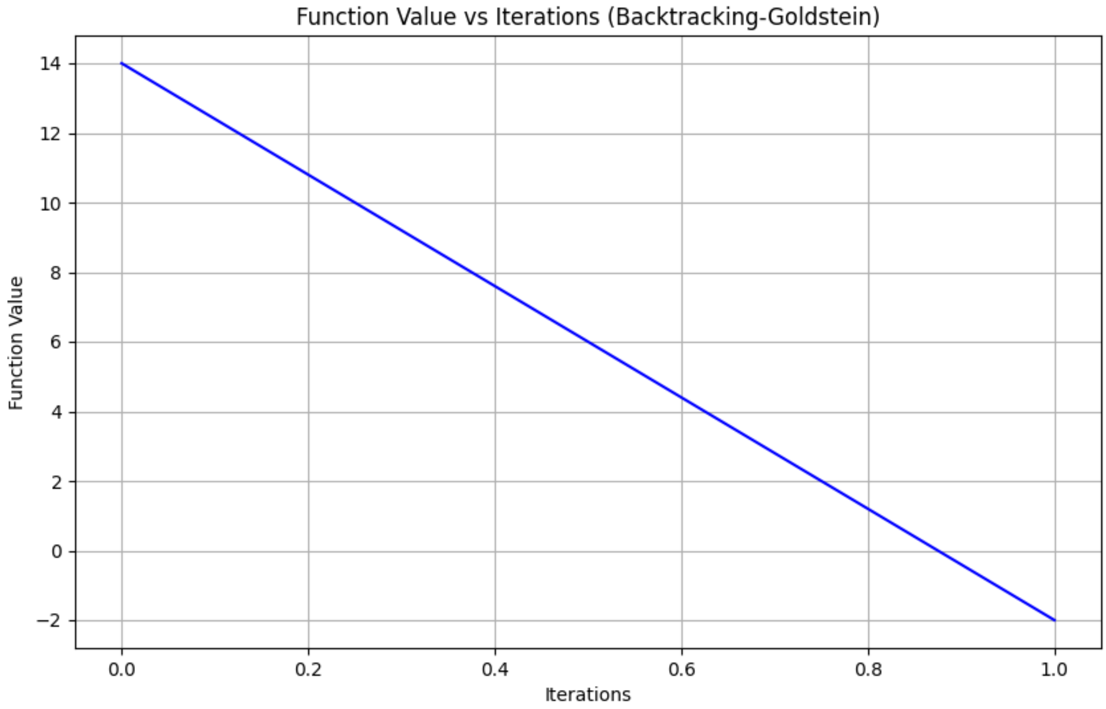
Backtracking-Armijo:

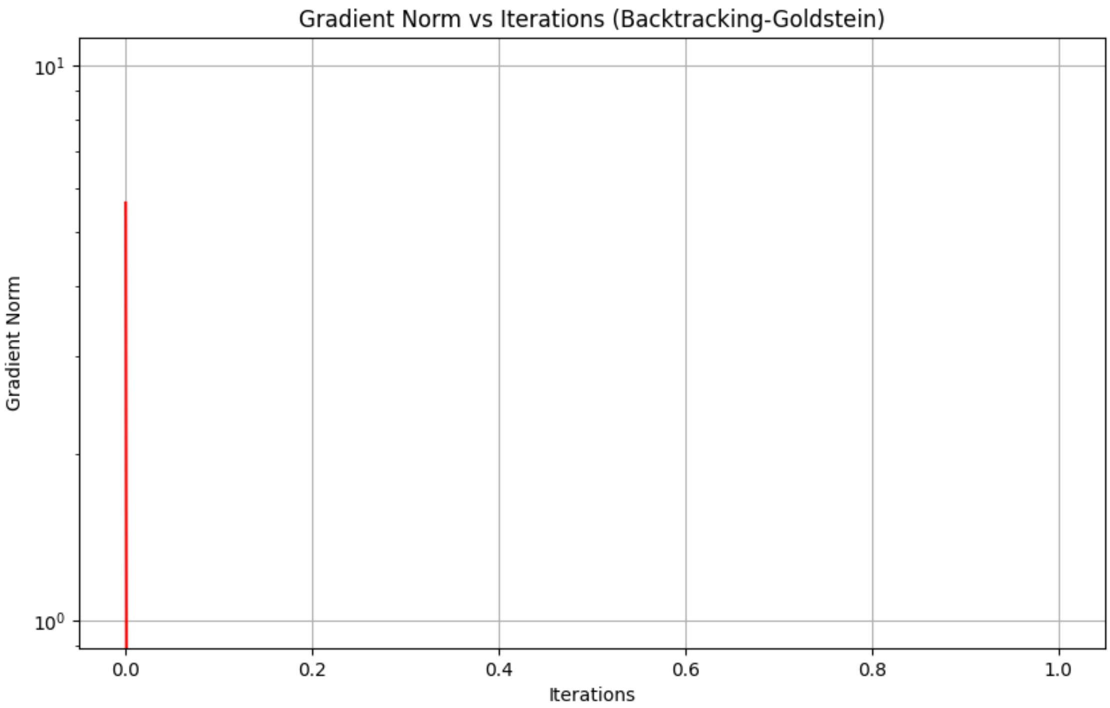




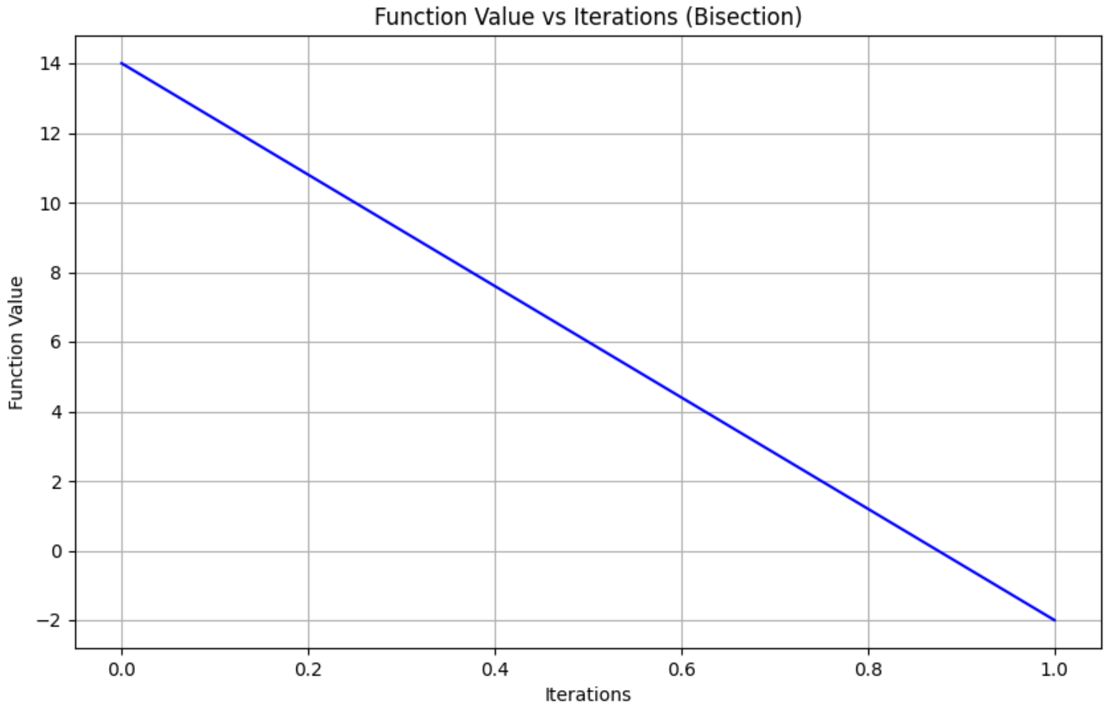
PROF

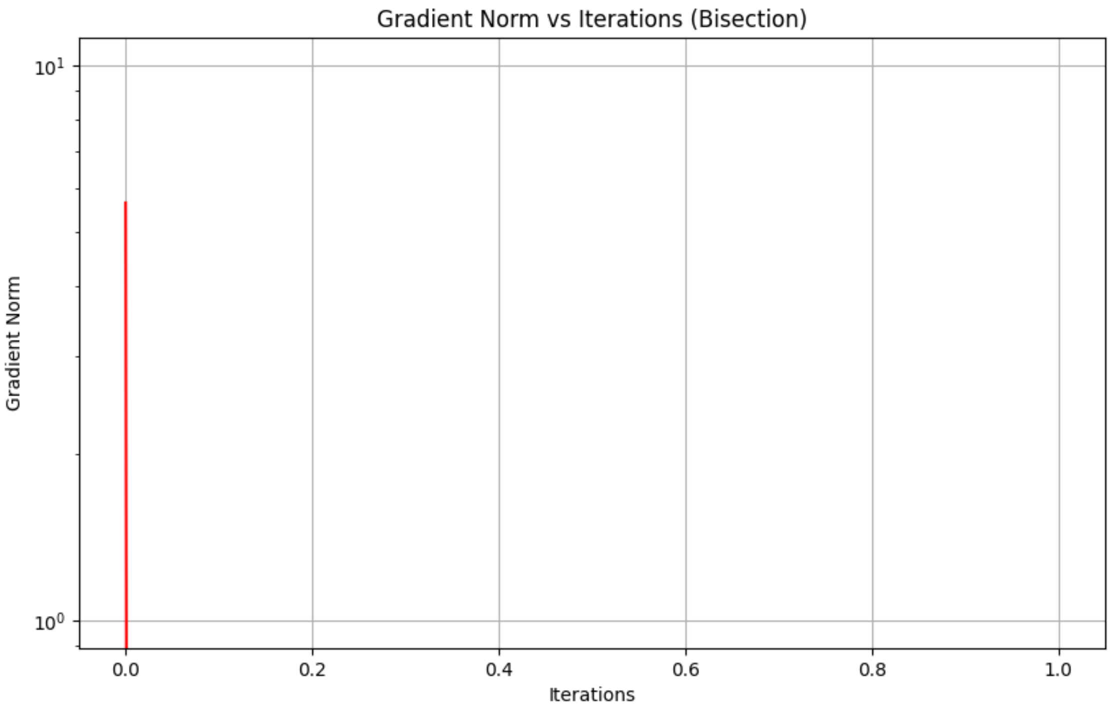
Backtracking-Goldstein:





Bisection:

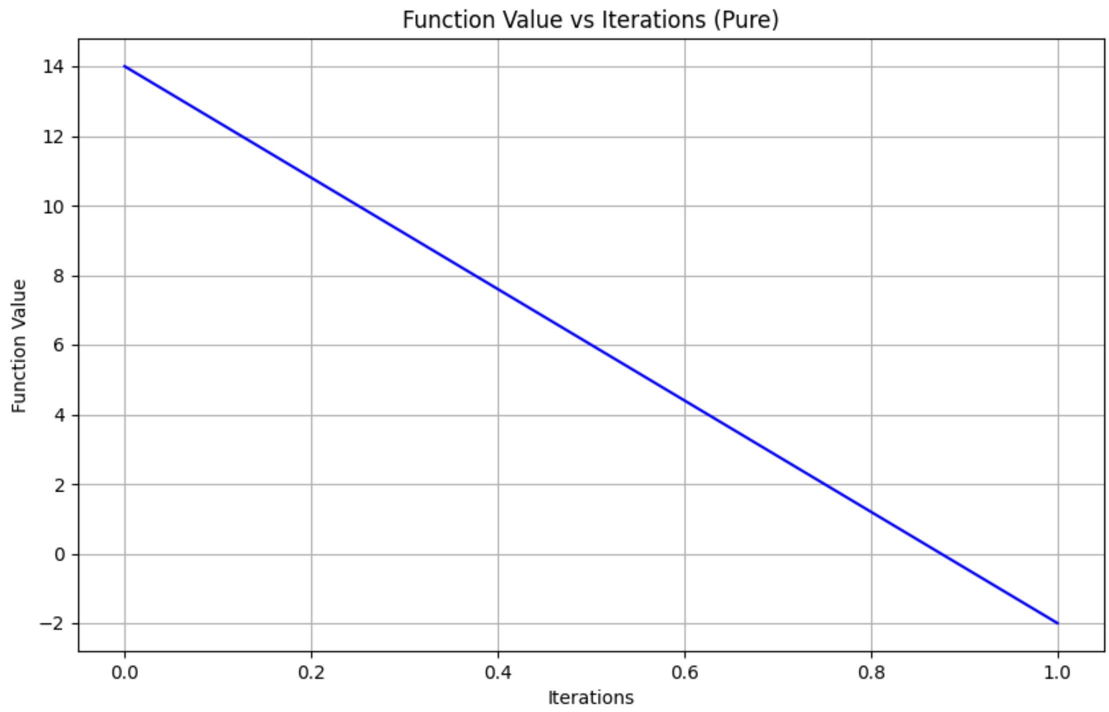


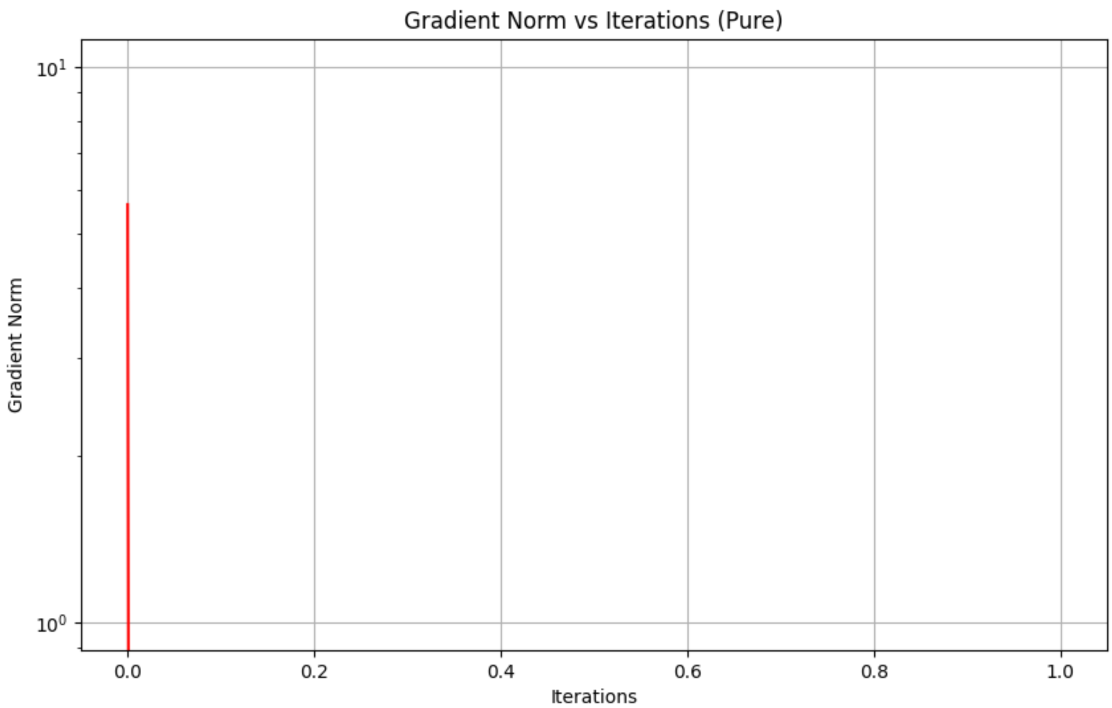


PROF

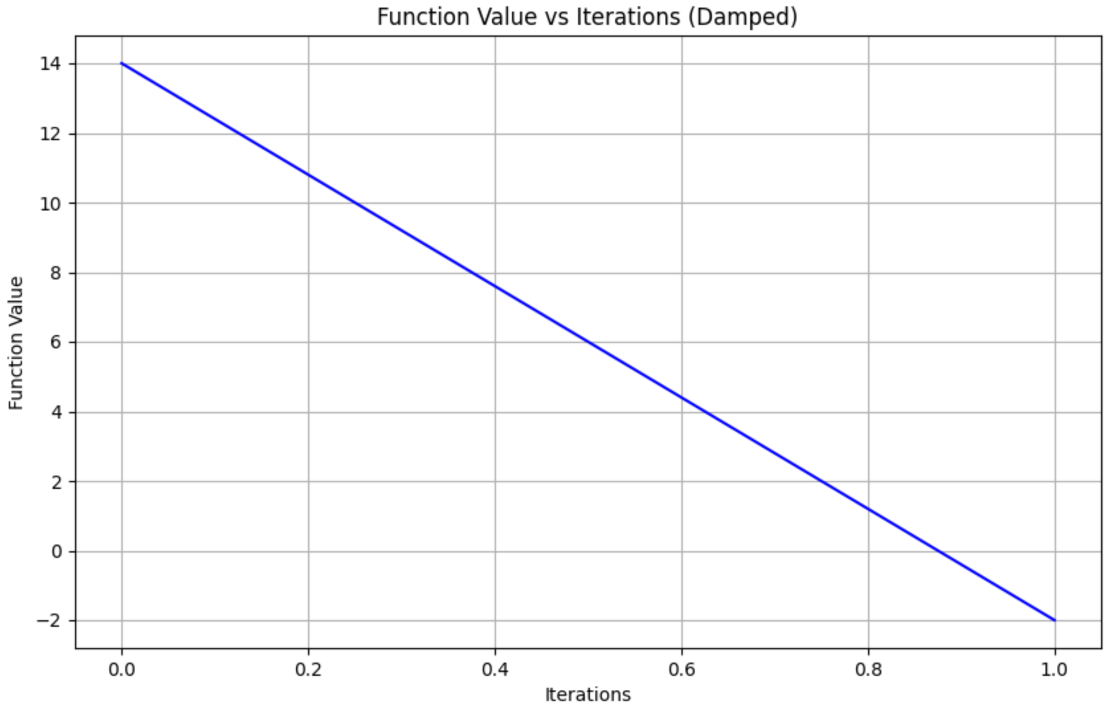
Newton's Methods:

Pure:

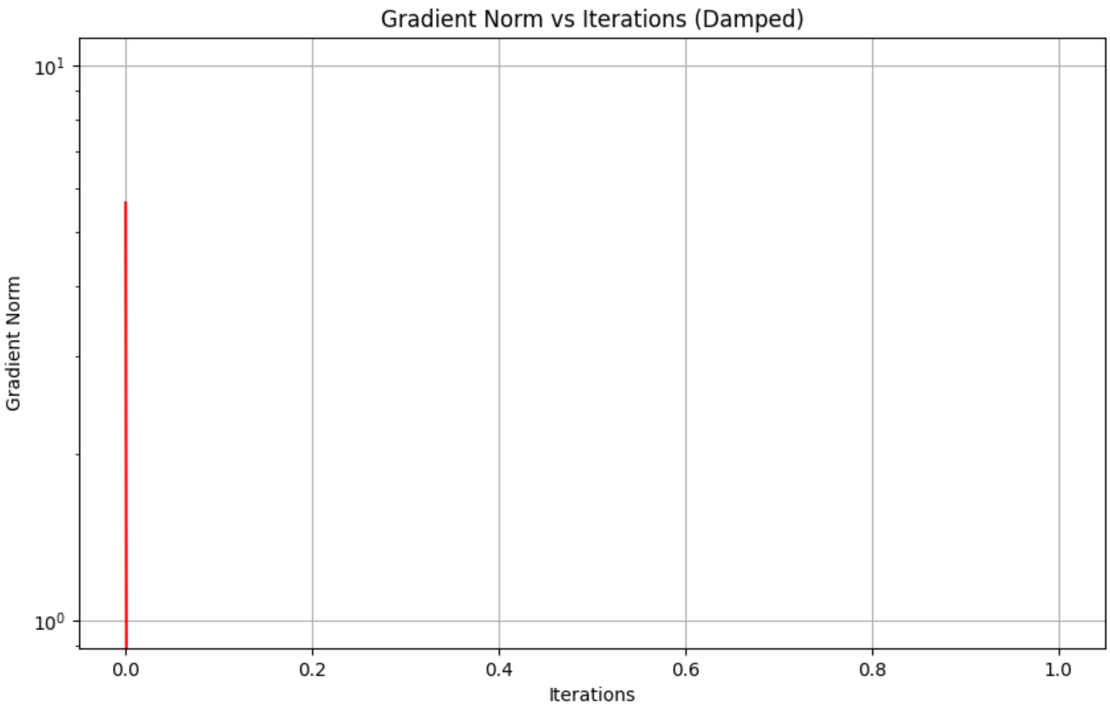

Function Value vs Iterations (Pure)
Function Value
Iterations
14
12
10
8
6
4
2
0
−2
0.0
0.2
0.4
0.6
0.8
1.0


Gradient Norm vs Iterations (Pure)
Gradient Norm
Iterations
10^1
10^0
0.0
0.2
0.4
0.6
0.8
1.0

Damped:

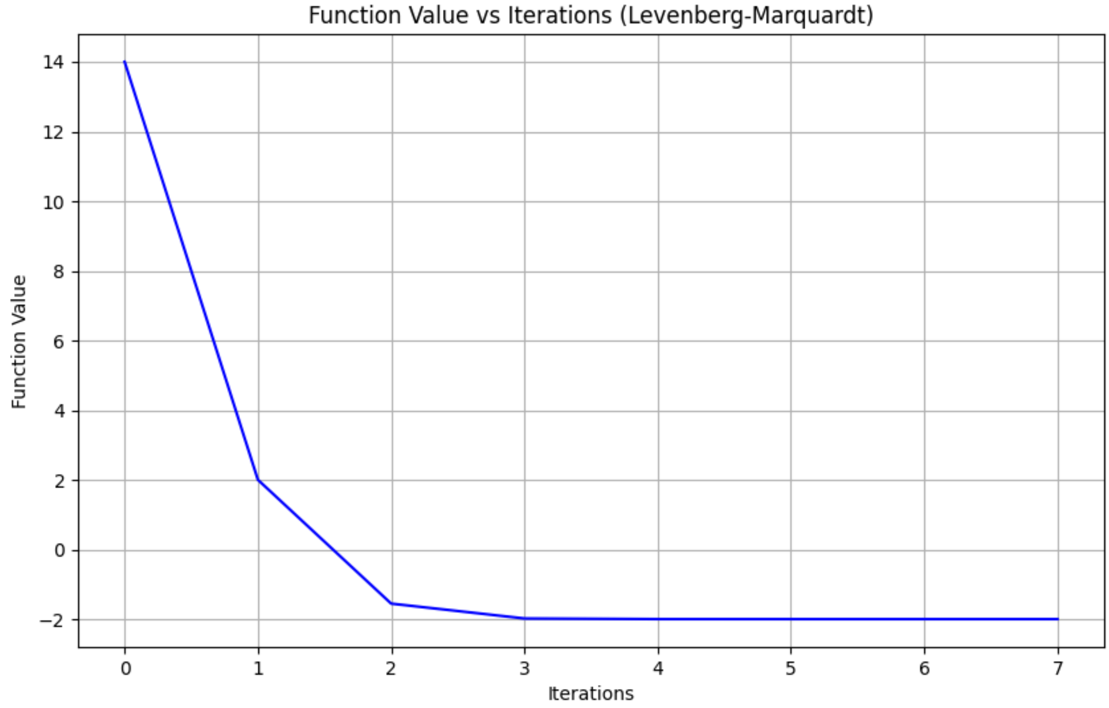


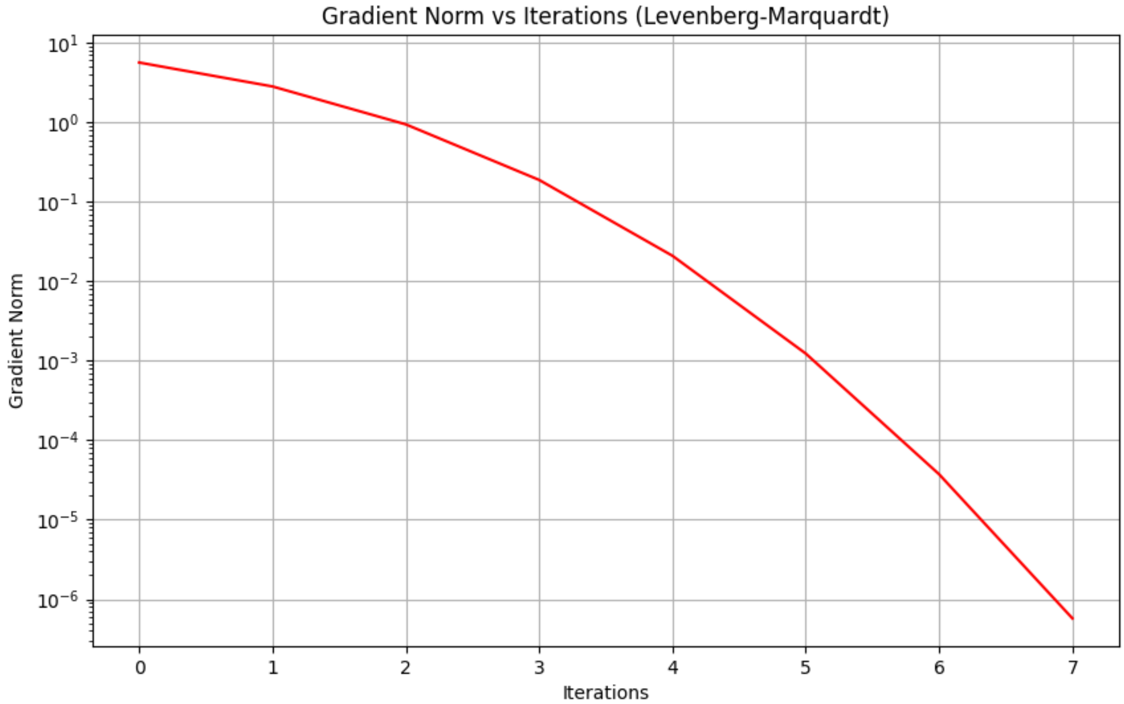
Function Value vs Iterations (Damped)
Function Value
Iterations
14
12
10
8
6
4
2
0
−2
0.0
0.2
0.4
0.6
0.8
1.0



Gradient Norm vs Iterations (Damped)
Gradient Norm
Iterations
10^1
10^0
0.0
0.2
0.4
0.6
0.8
1.0

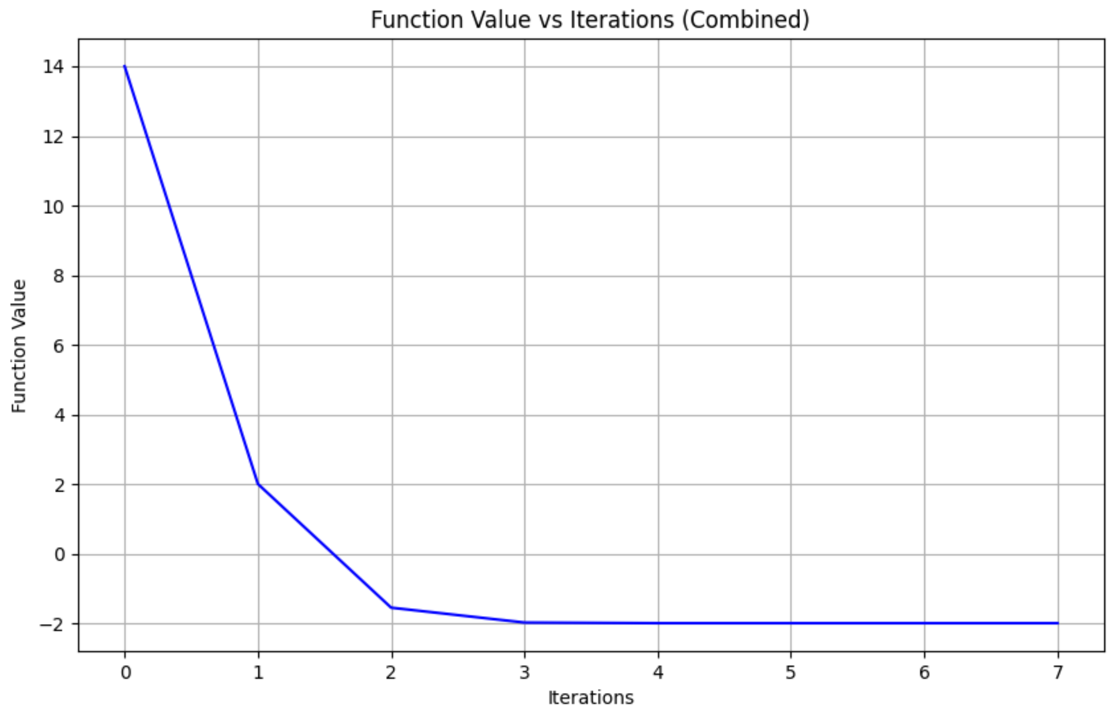
Levenberg-Marquardt:

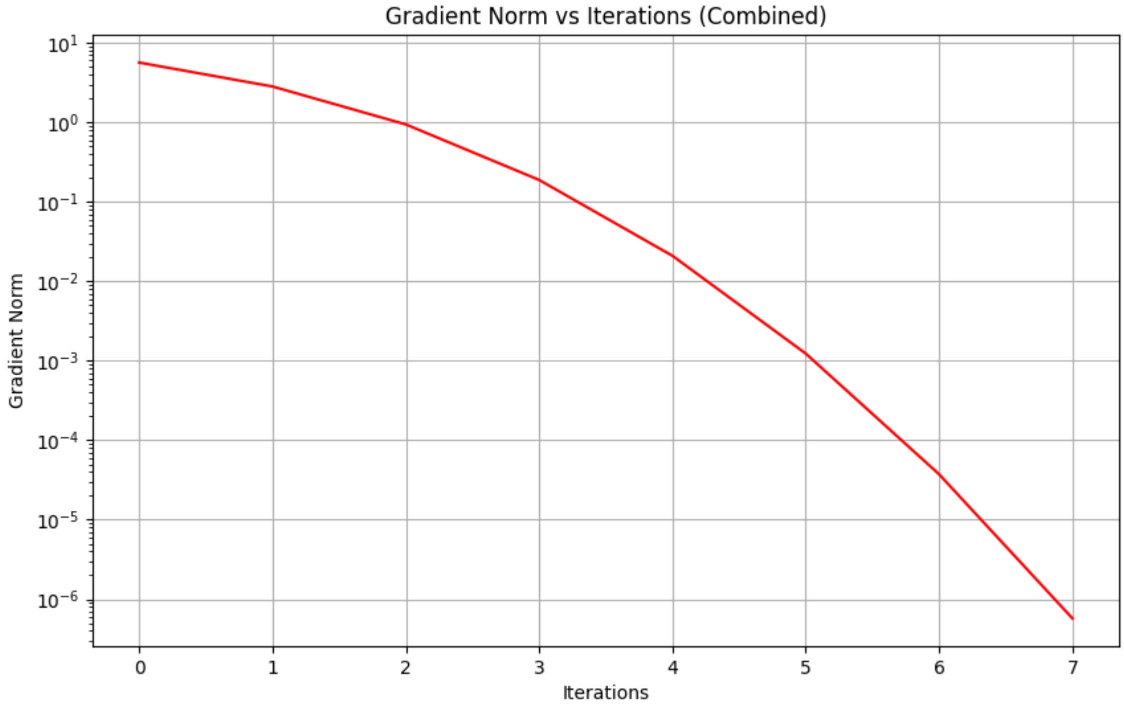




PROF

Combined:





Similarly, for other functions, you can check the plots directory.

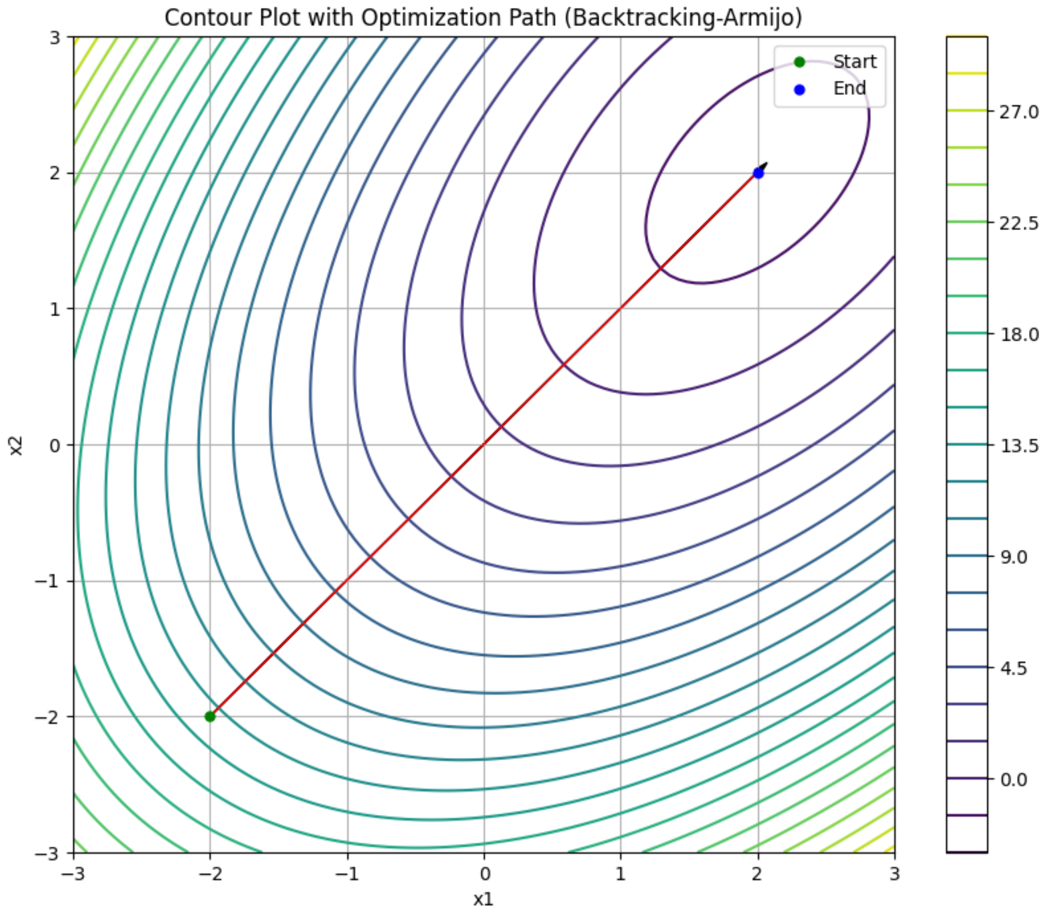
5. Contour Plots for 2D Functions

5.1 Trid Function

5.1.1 Initial Point: [-2.0, -2.0]

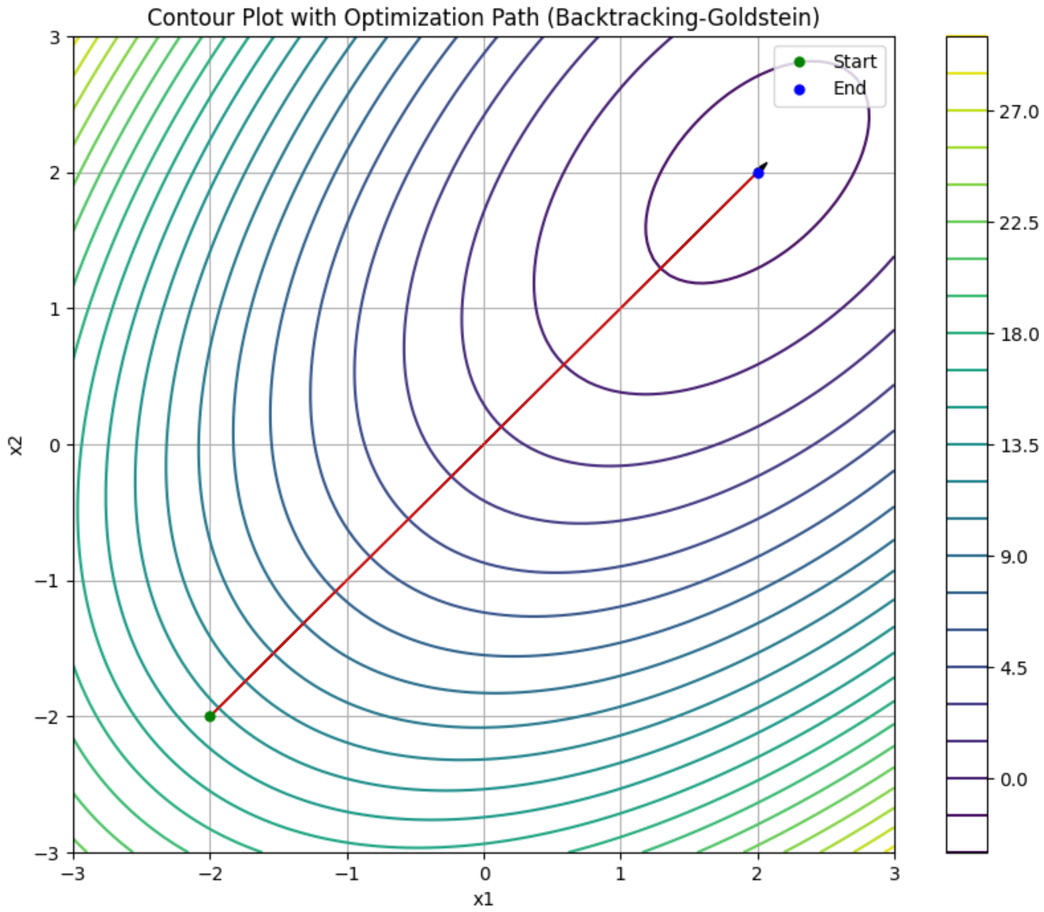
Steepest Descent Methods:

Backtracking-Armijo:

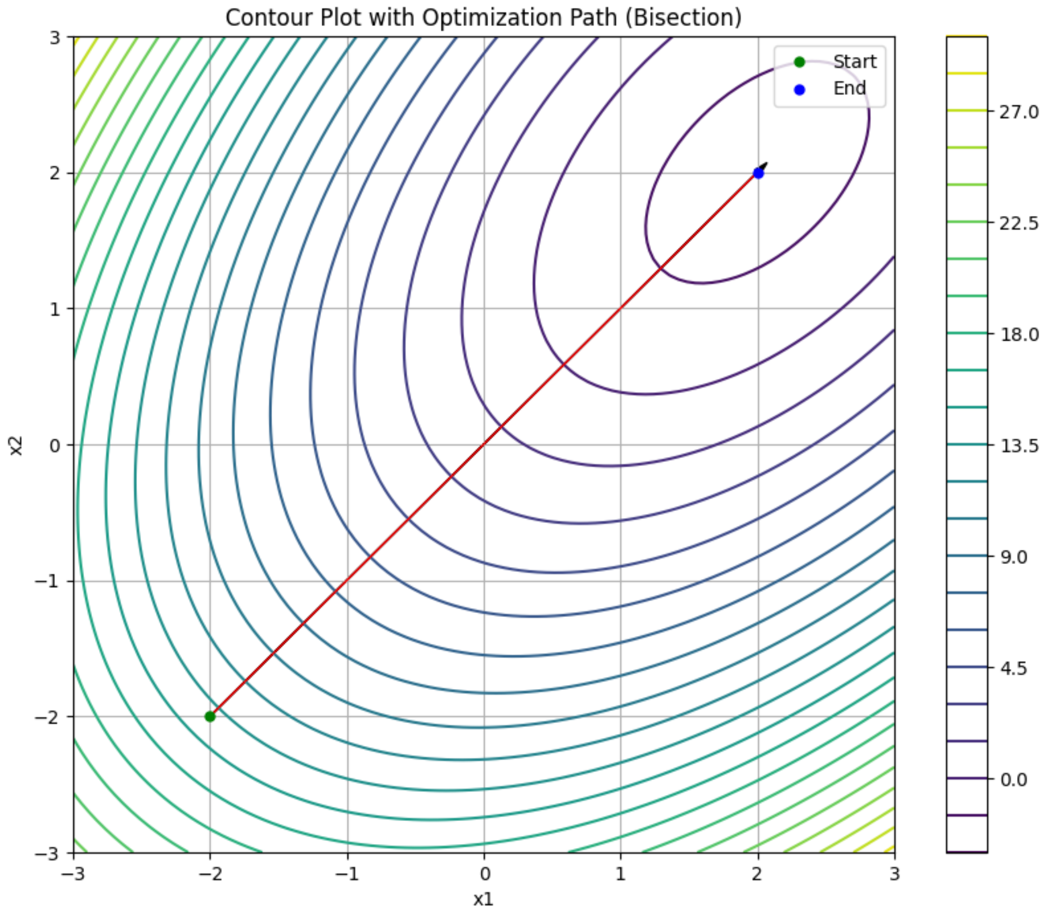


PROF

Backtracking-Goldstein:



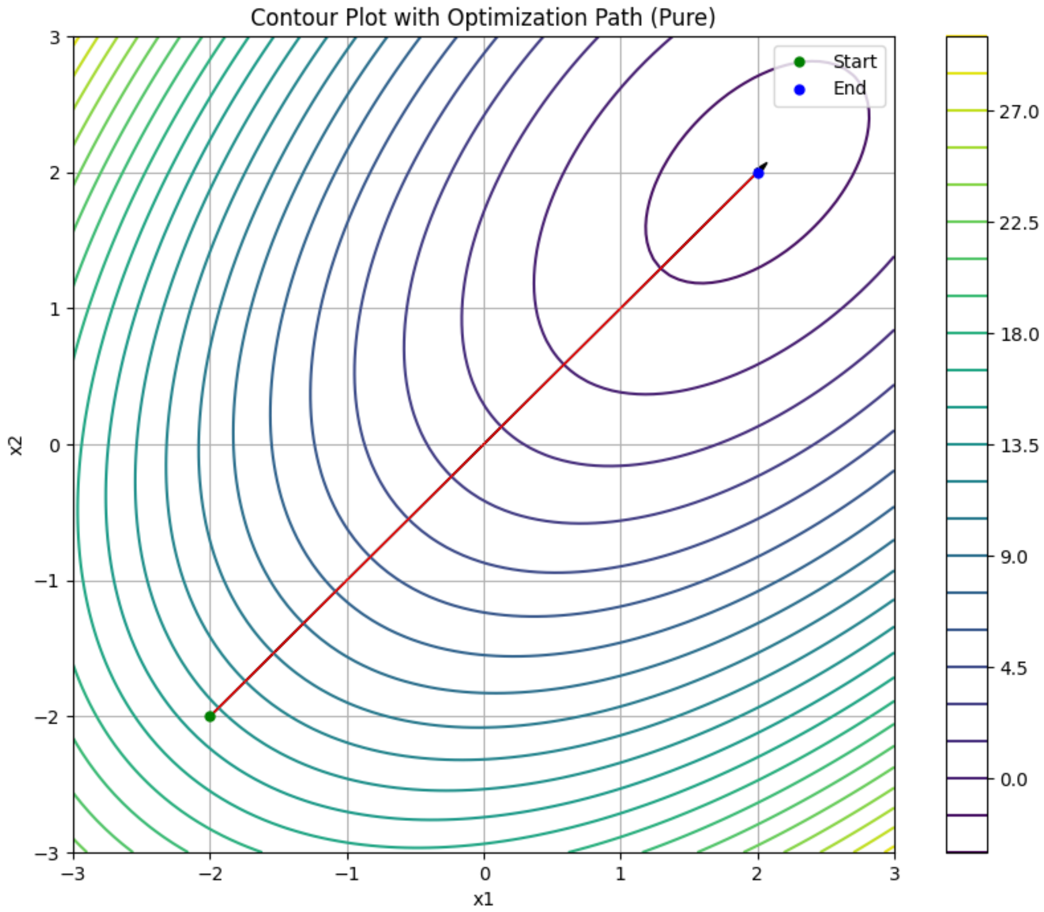
Bisection:



Newton's Methods:

PROF

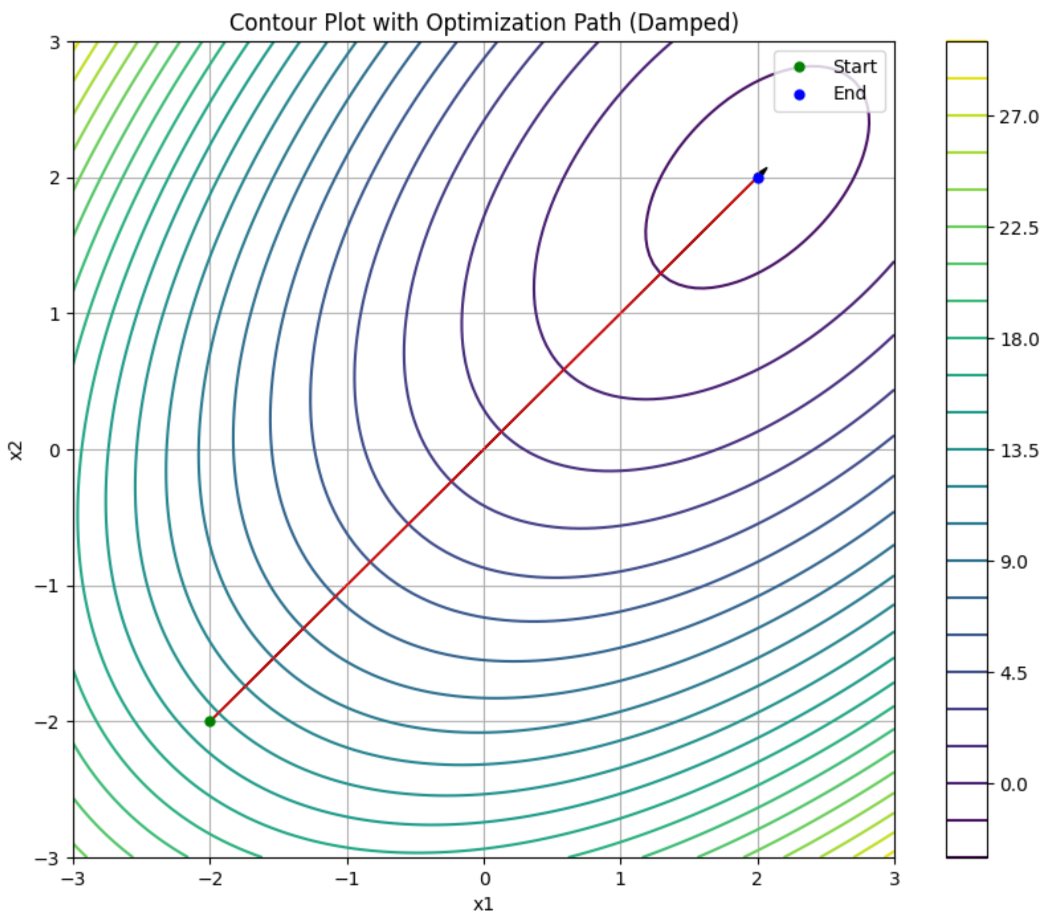
Pure:



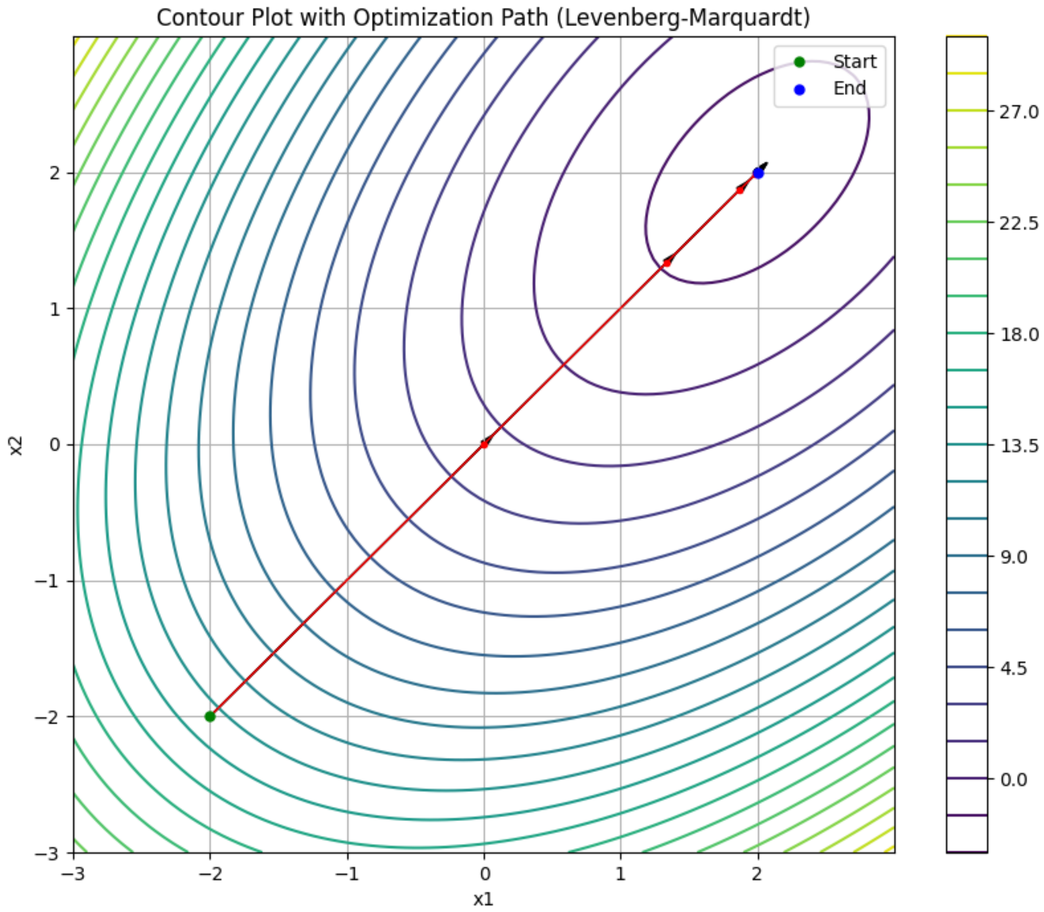
Contour Plot with Optimization Path (Pure)
Start
End
x1
x2
−3
−2
−1
0
1
2
3
27.0
22.5
18.0
13.5
9.0
4.5
0.0

PROF

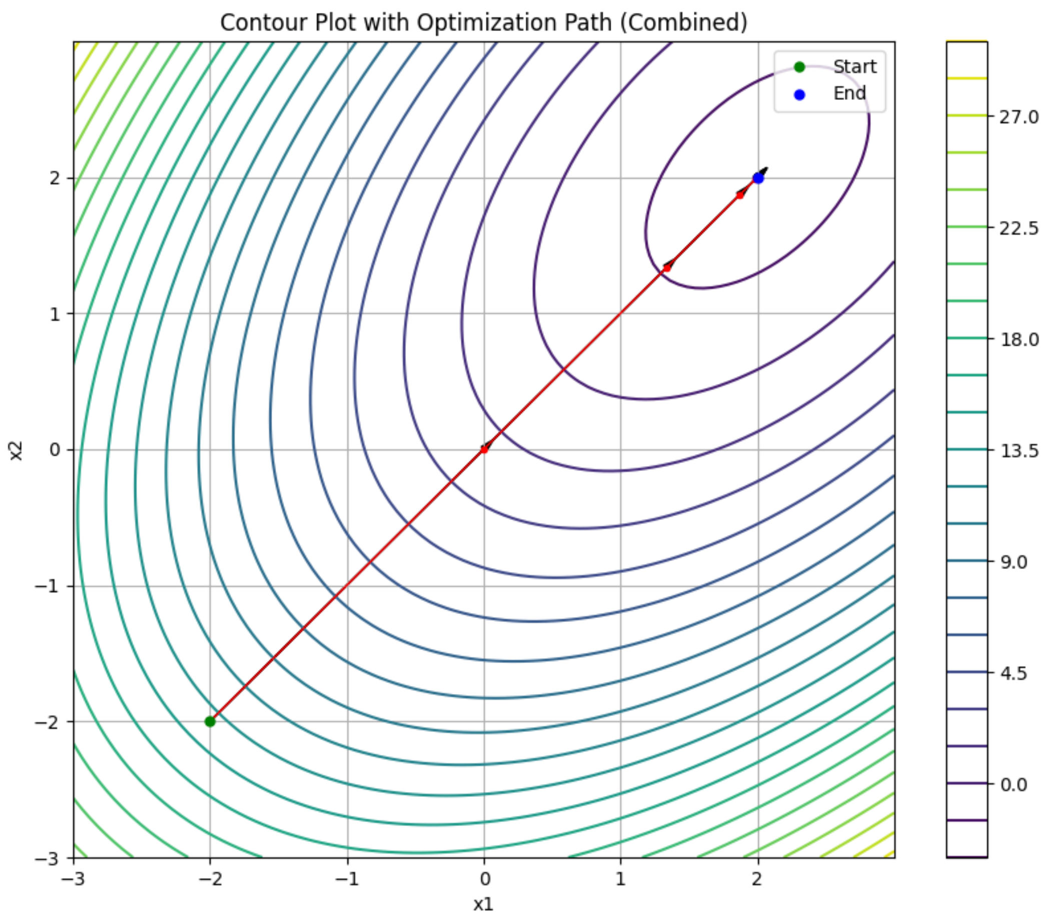
Damped:


Contour Plot with Optimization Path (Damped)
Start
End
x1
x2
27.0
22.5
18.0
13.5
9.0
4.5
0.0

Levenberg-Marquardt:



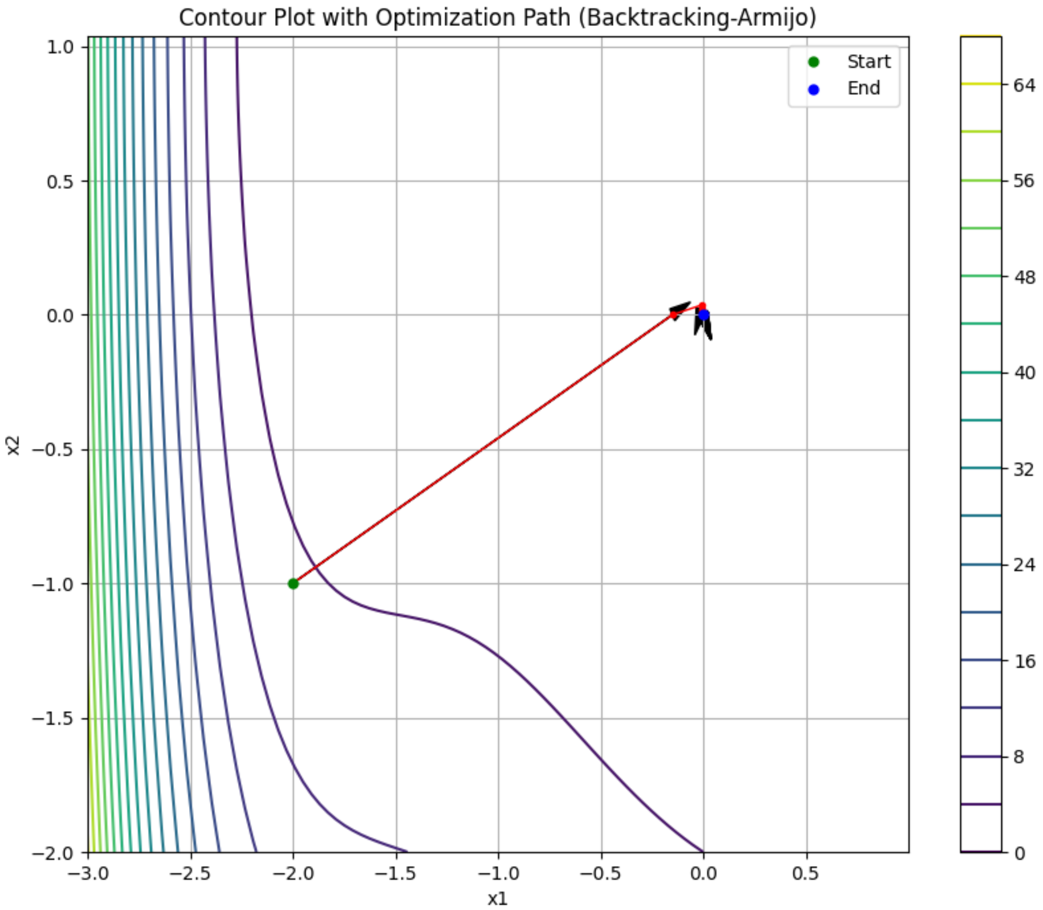
Combined:



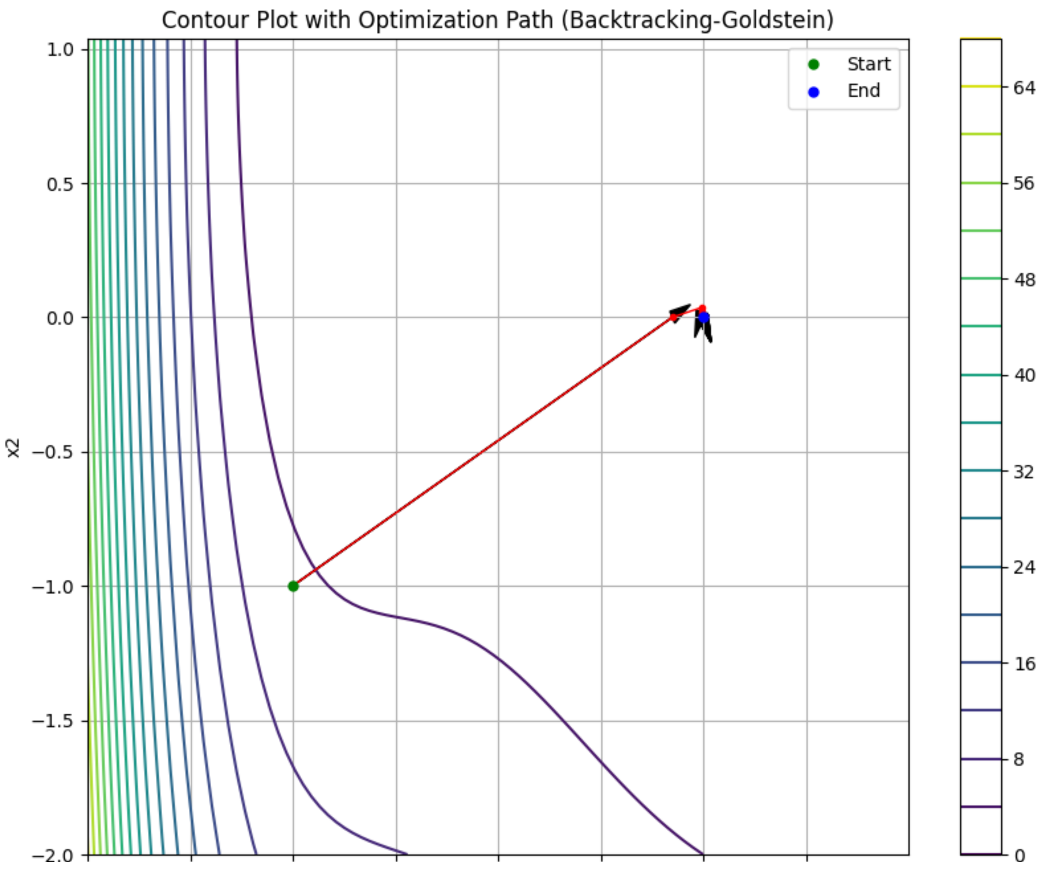
5.2 Three Hump Camel Function

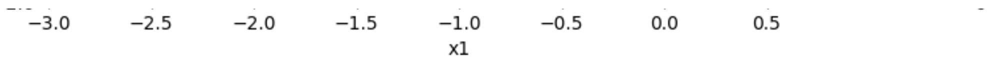
5.2.1 Initial Point: [-2.0, -1.0]

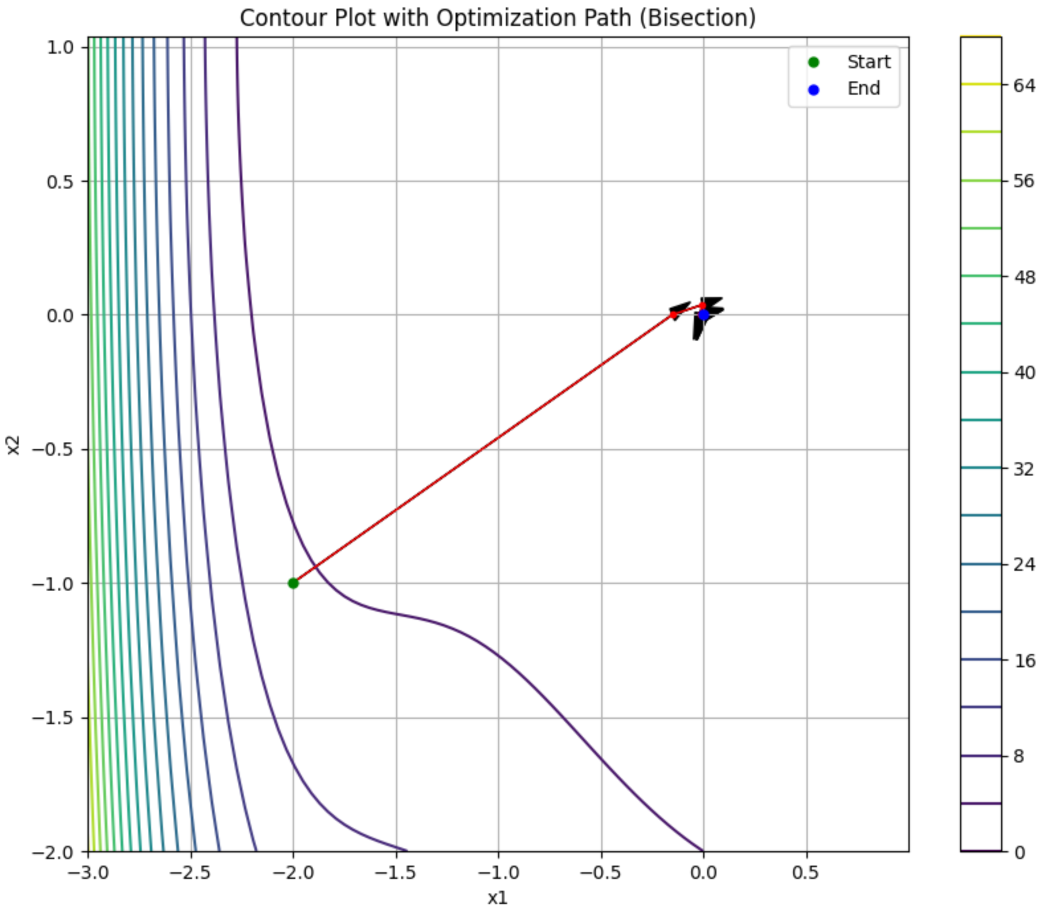
PROF


Contour Plot with Optimization Path (Backtracking-Armijo)
Start
End
x1
x2

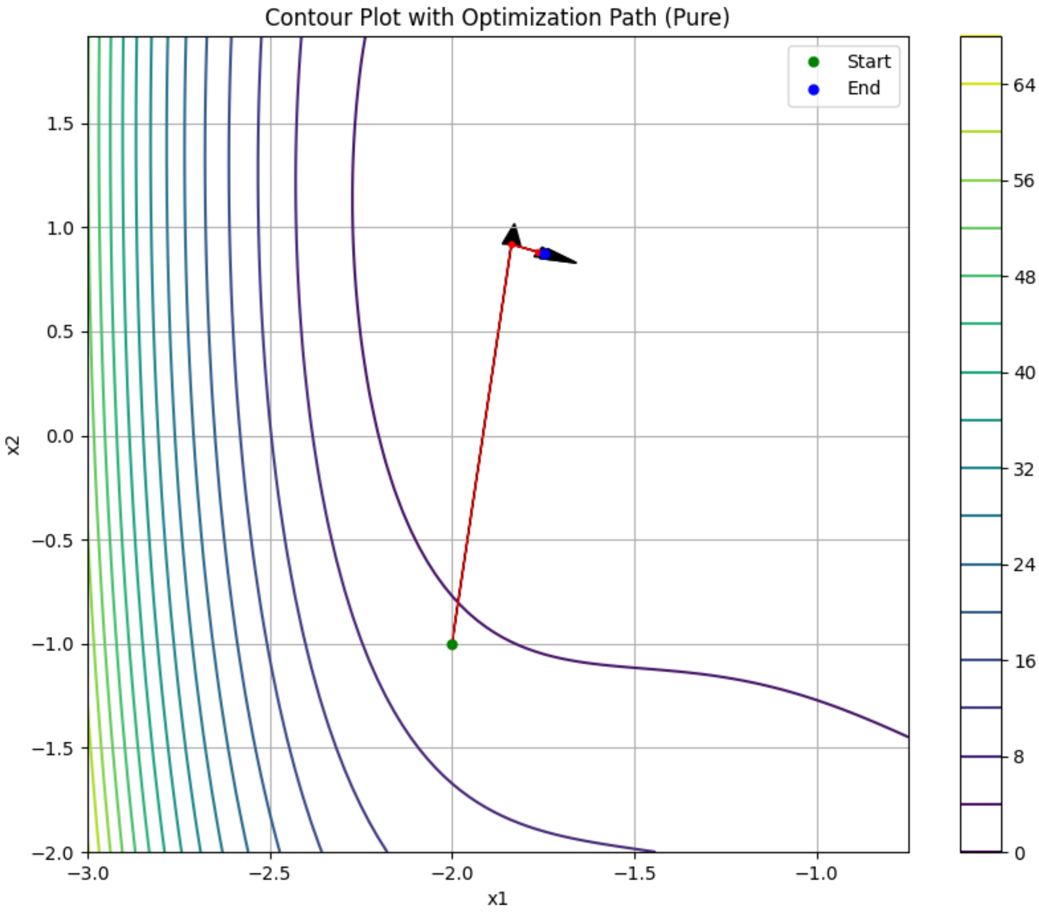
PROF

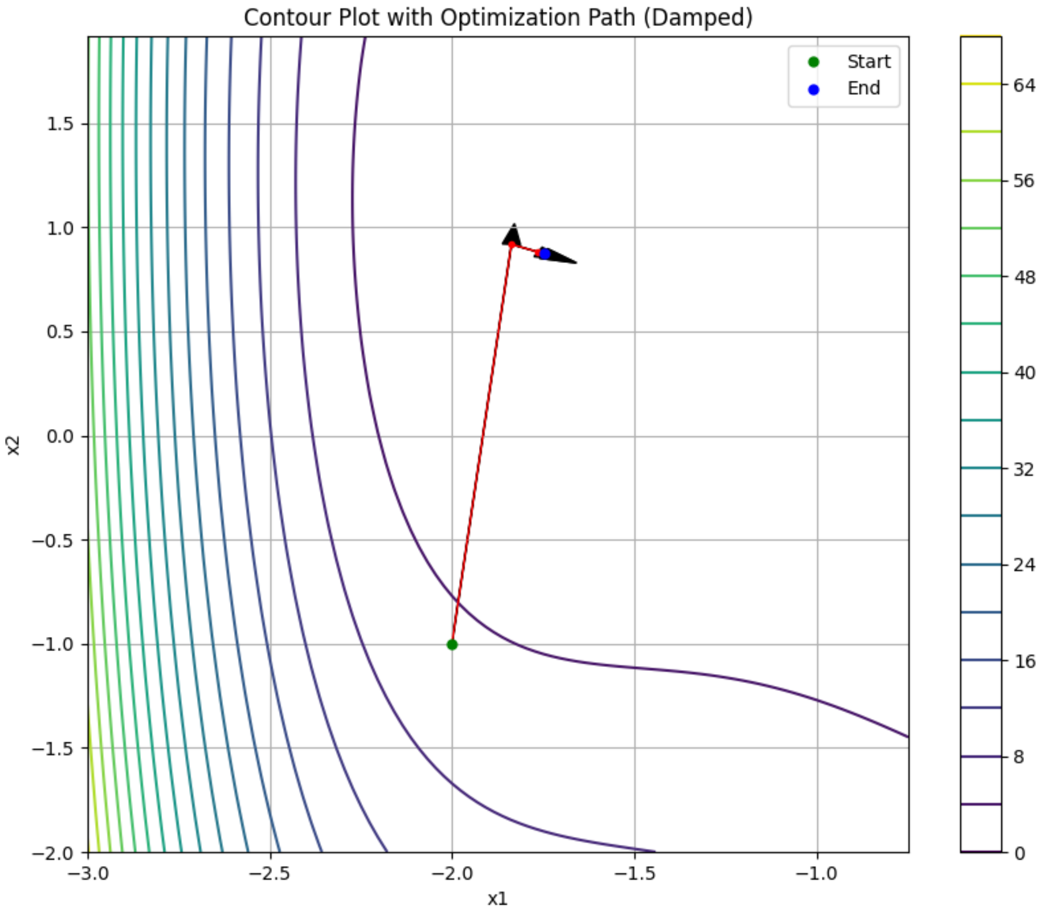

Contour Plot with Optimization Path (Backtracking-Goldstein)
Start
End
x2


−3.0
−2.5
−2.0
−1.5
−1.0
−0.5
0.0
0.5
x1

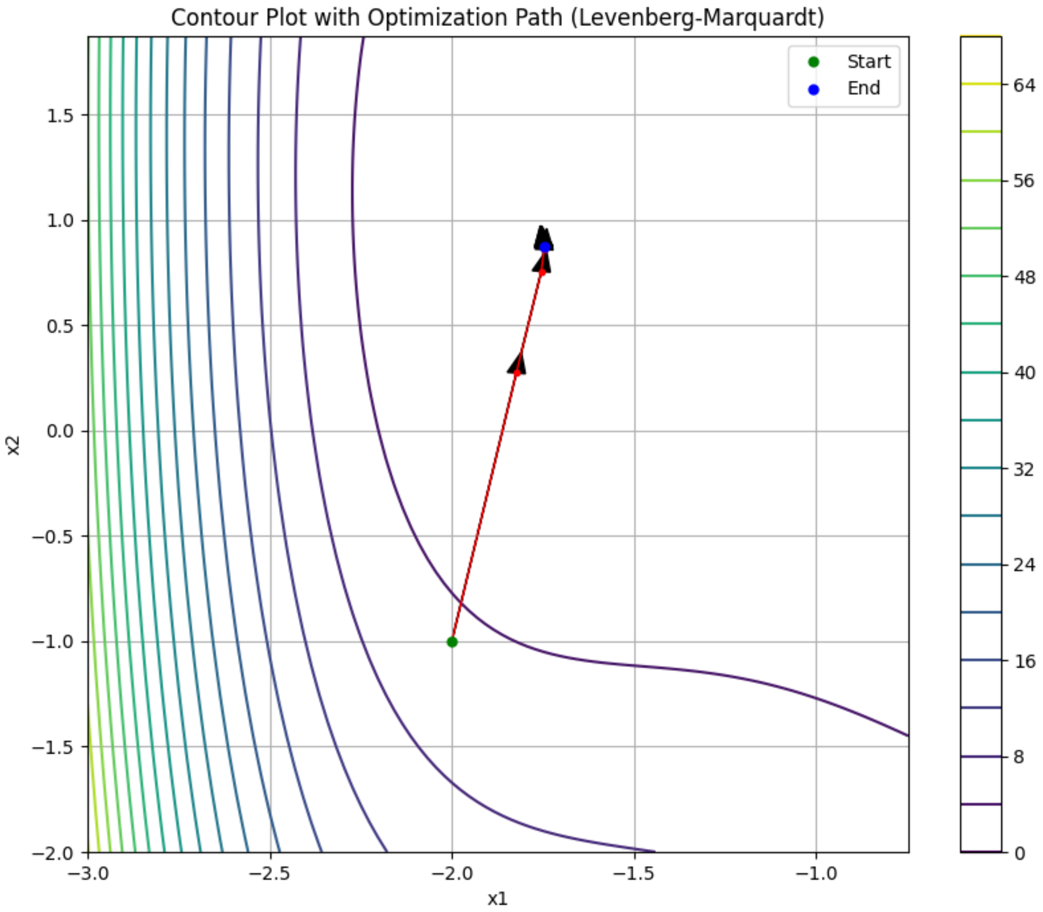

Contour Plot with Optimization Path (Bisection)
Start
End
x1
x2
−3.0
−2.5
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
64
56
48
40
32
24
16
8
0

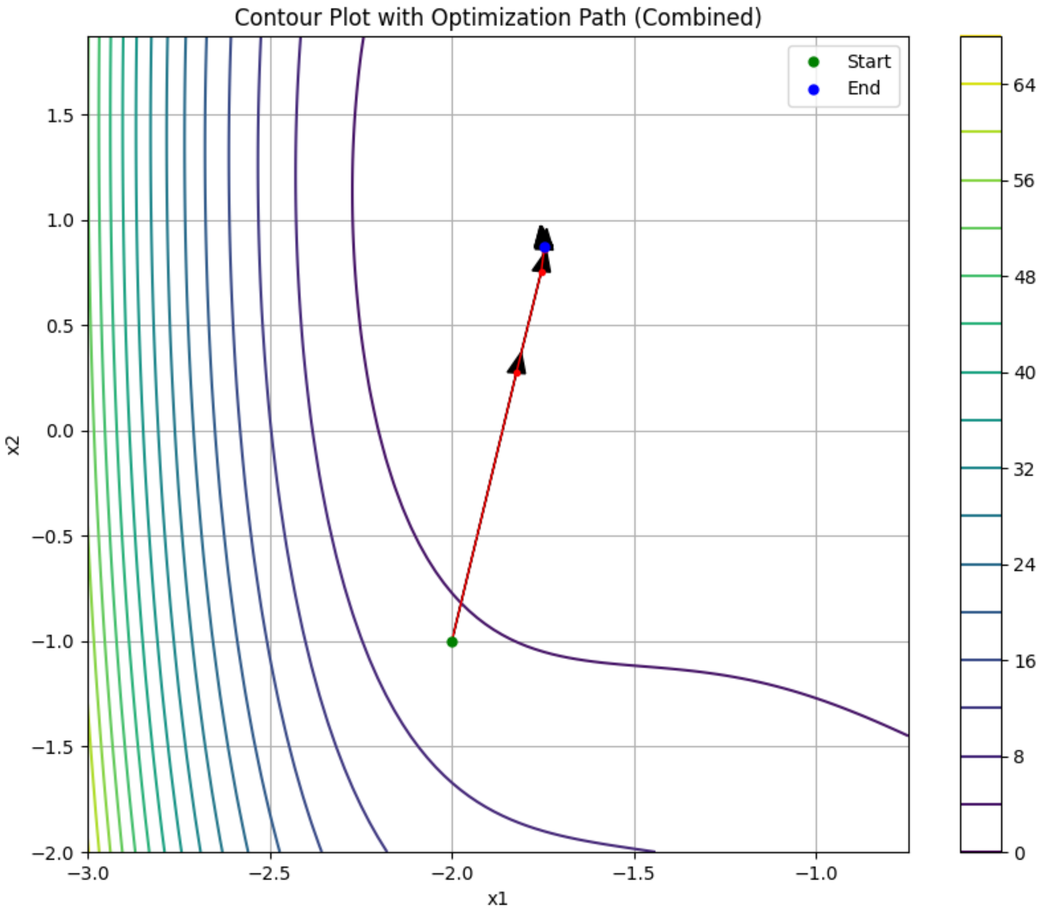
PROF


Contour Plot with Optimization Path (Pure)
Start
End
x1
x2
−3.0
−2.5
−2.0
−1.5
−1.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
64
56
48
40
32
24
16
8
0



Contour Plot with Optimization Path (Damped)
Start
End
x1
x2
−3.0
−2.5
−2.0
−1.5
−1.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
64
56
48
40
32
24
16
8
0

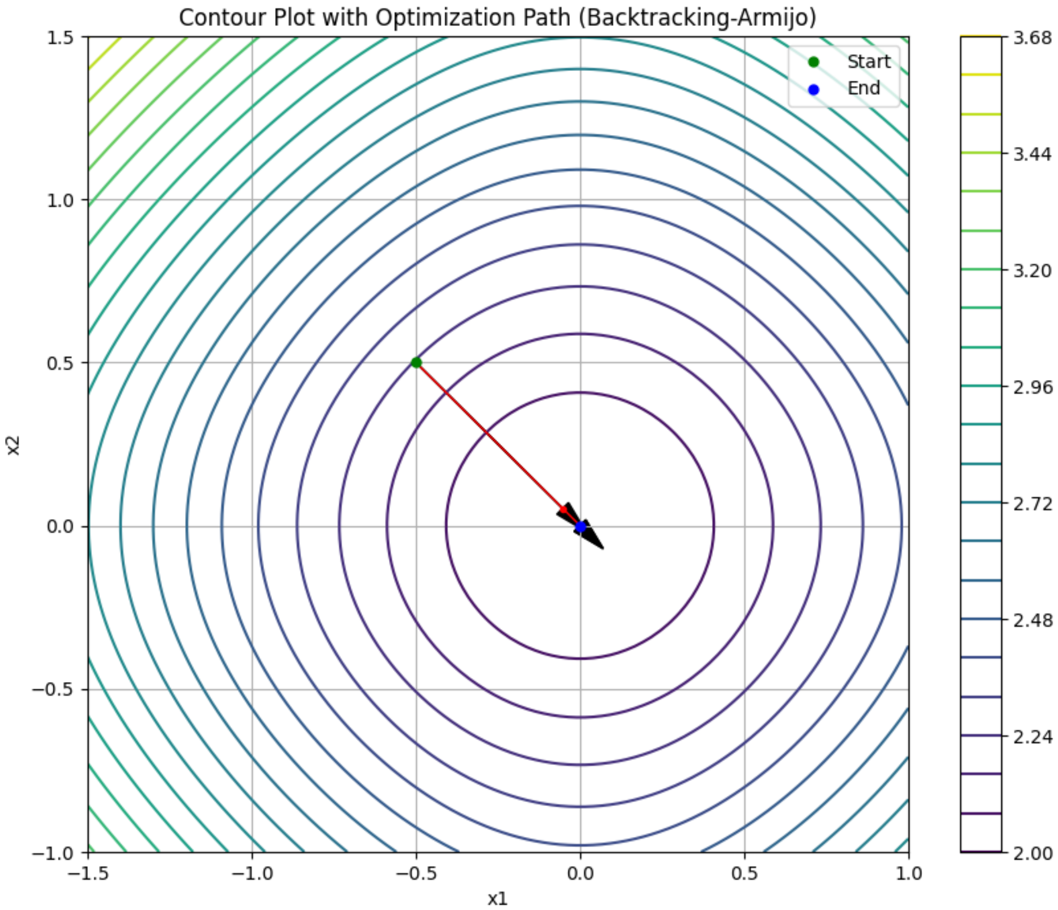

Contour Plot with Optimization Path (Levenberg-Marquardt)
Start
End
x1
x2
−3.0
−2.5
−2.0
−1.5
−1.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
64
56
48
40
32
24
16
8
0



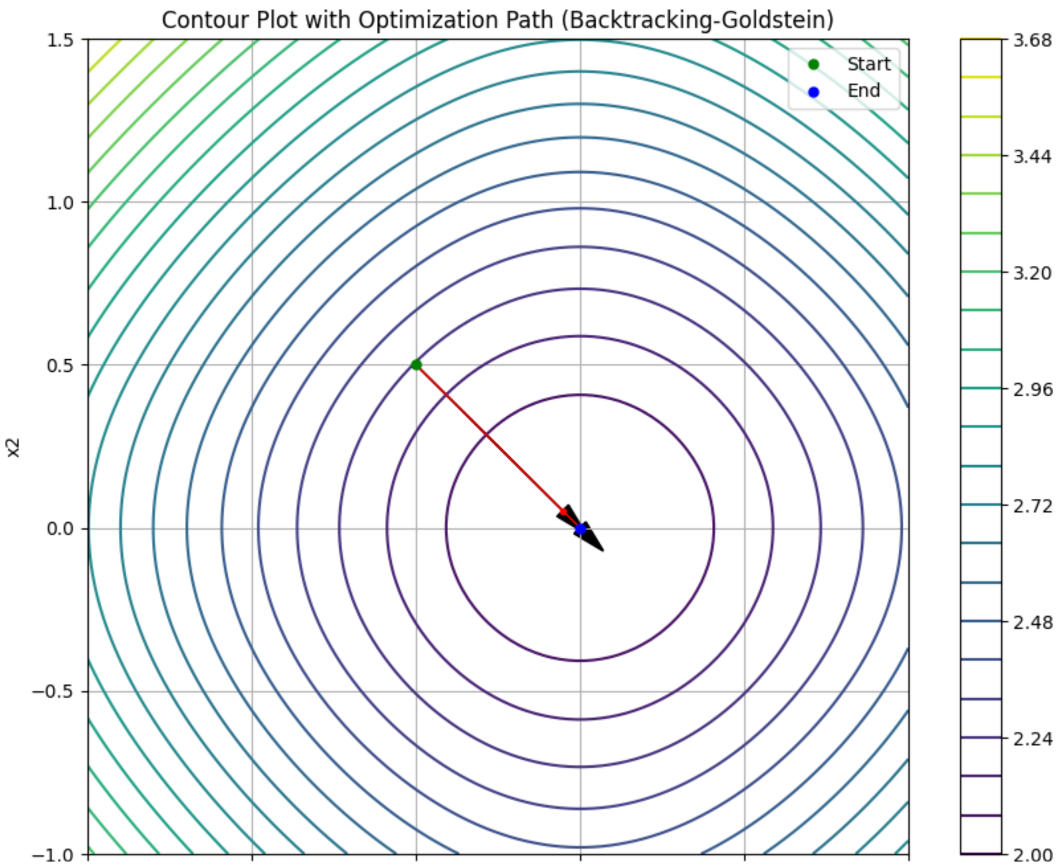
5.3 Root of Square Function (func_1)

5.3.1 Initial Point: [-.5, .5]

PROF

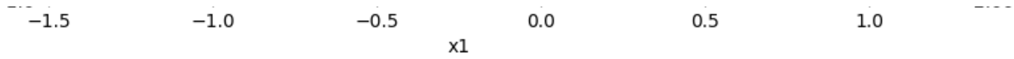


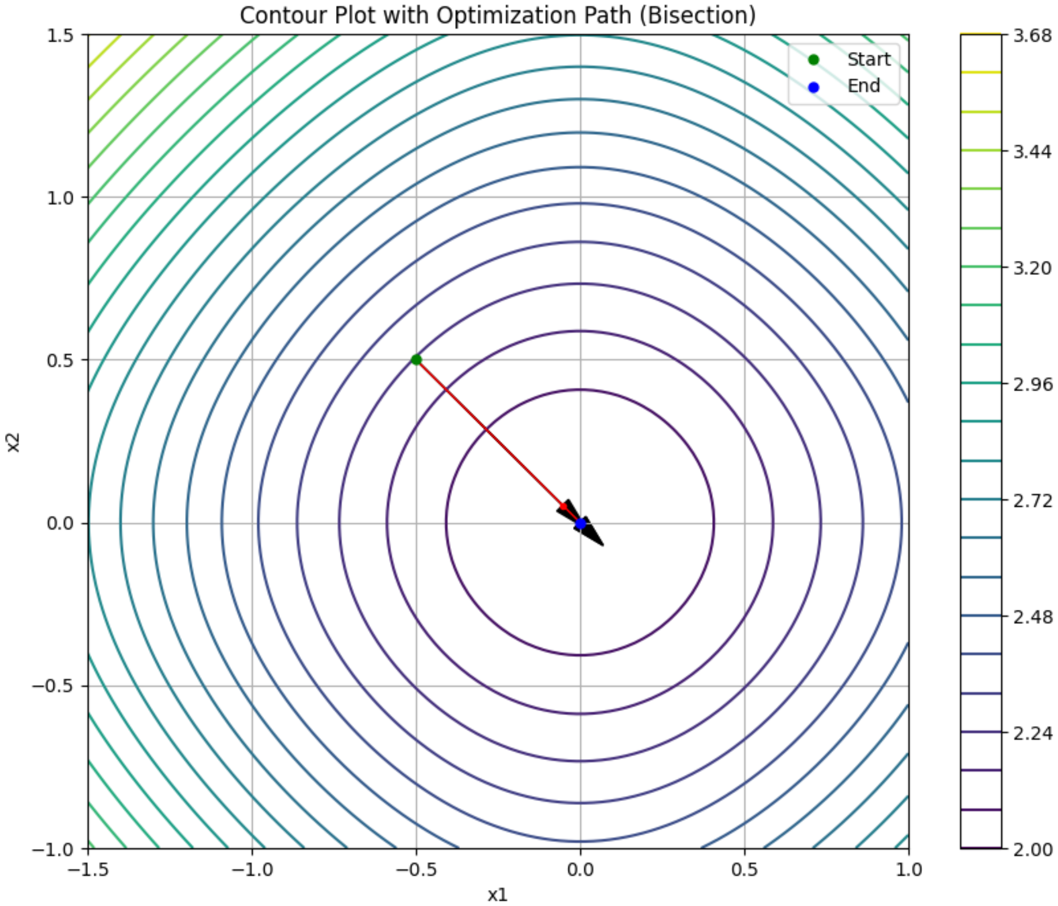
Contour Plot with Optimization Path (Backtracking-Armijo)
Start
End
x1
x2
3.68
3.44
3.20
2.96
2.72
2.48
2.24
2.00



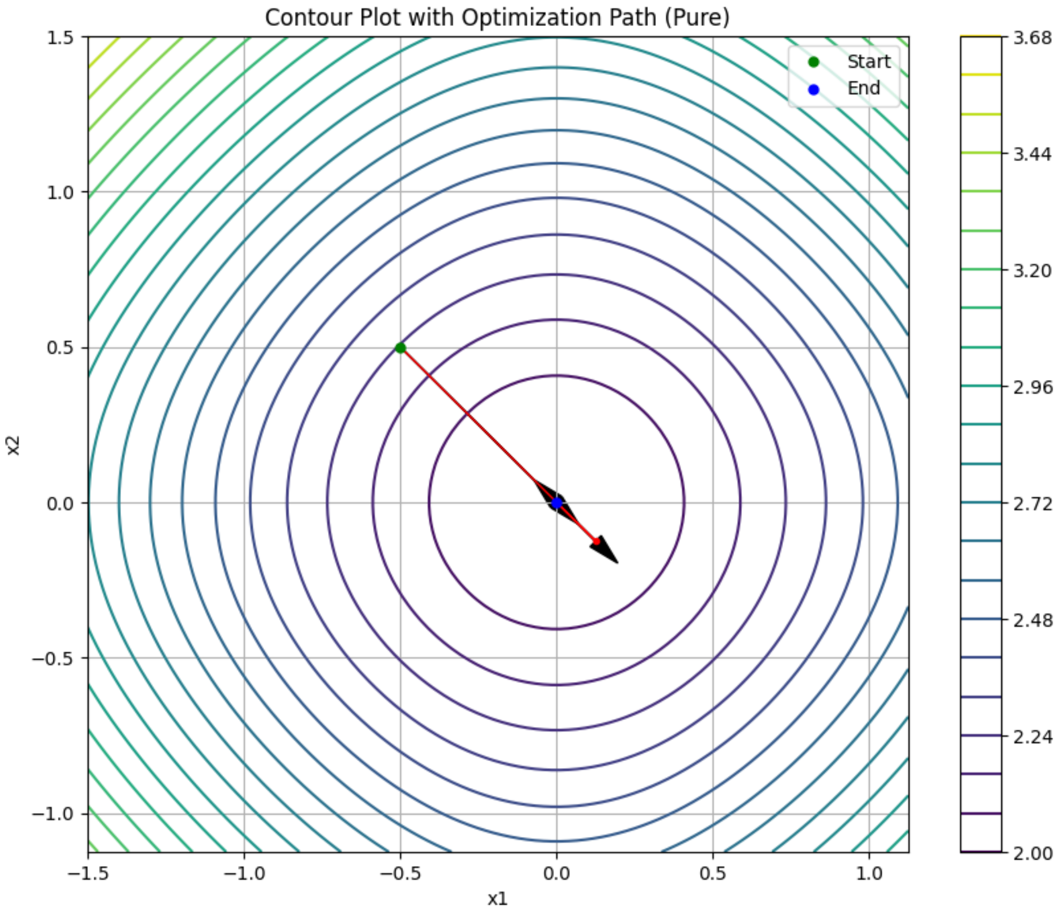
Contour Plot with Optimization Path (Backtracking-Goldstein)
Start
End
x2
3.68
3.44
3.20
2.96
2.72
2.48
2.24
2.00

PROF

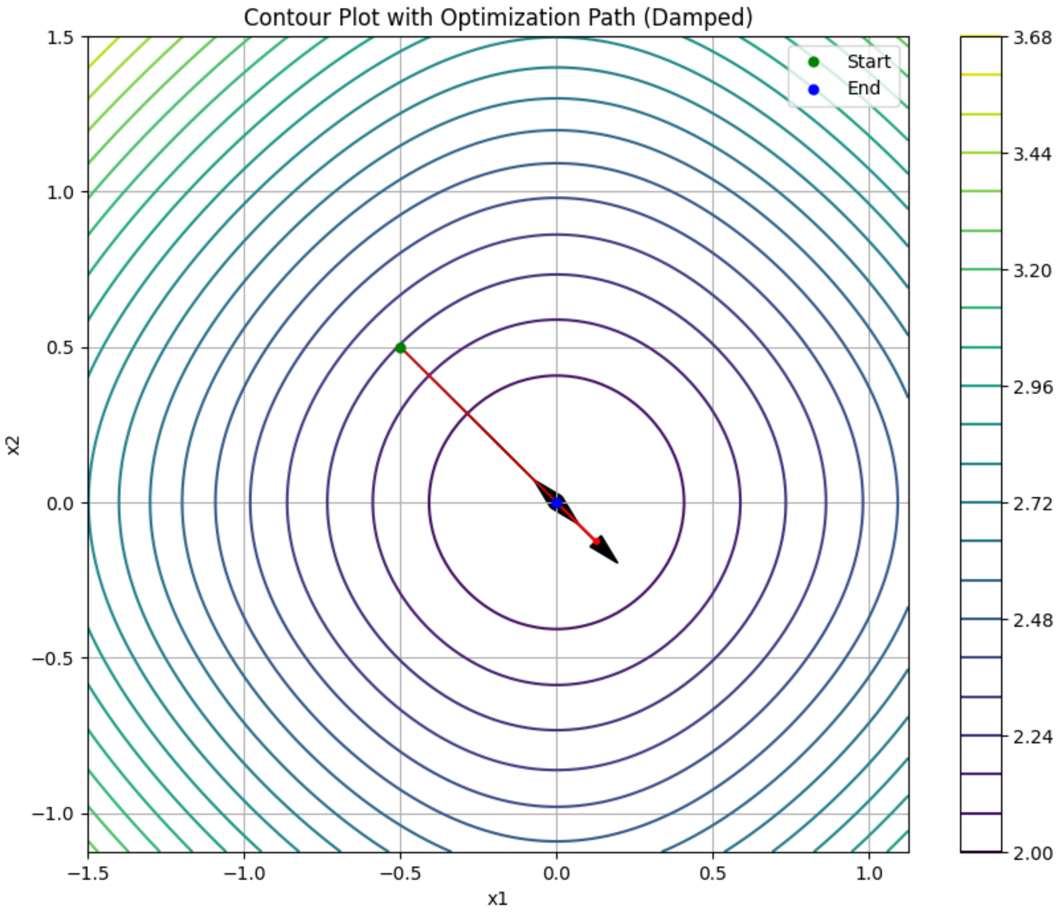

−1.5
−1.0
−0.5
0.0
0.5
1.0
x1

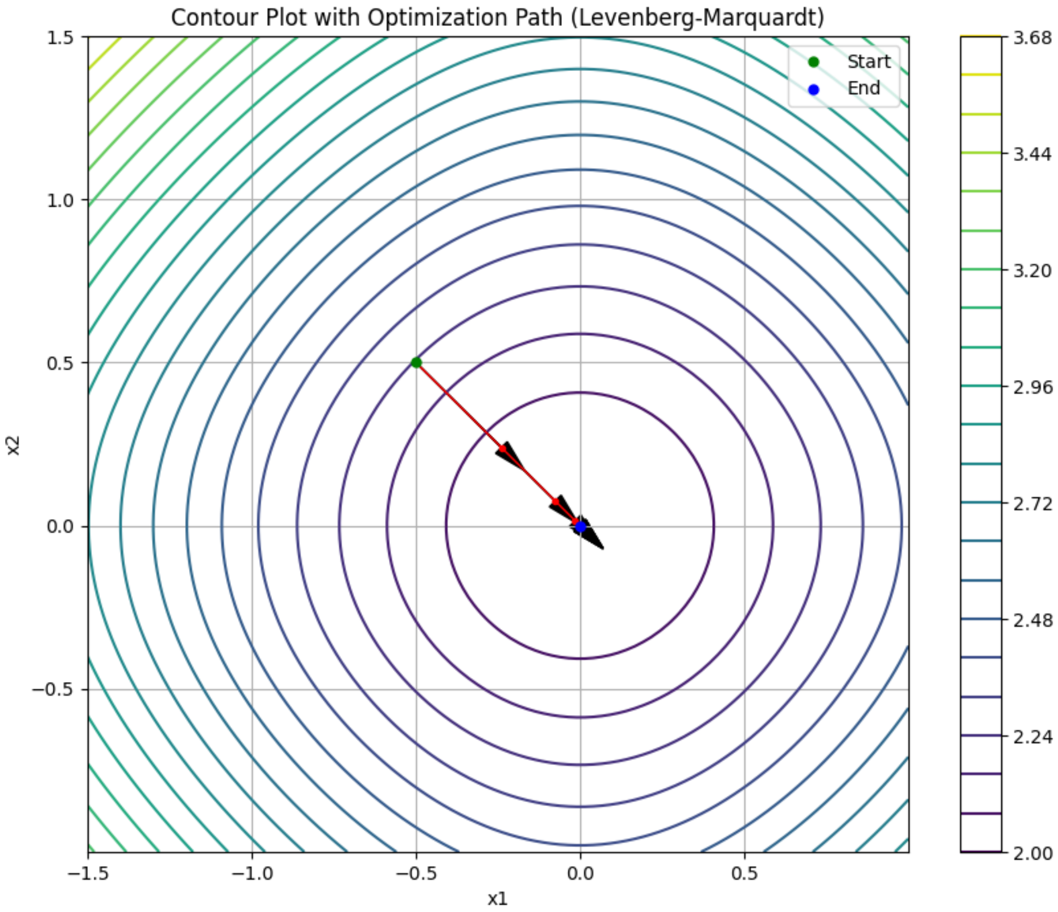

Contour Plot with Optimization Path (Bisection)
Start
End
x1
x2
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
3.68
3.44
3.20
2.96
2.72
2.48
2.24
2.00

PROF

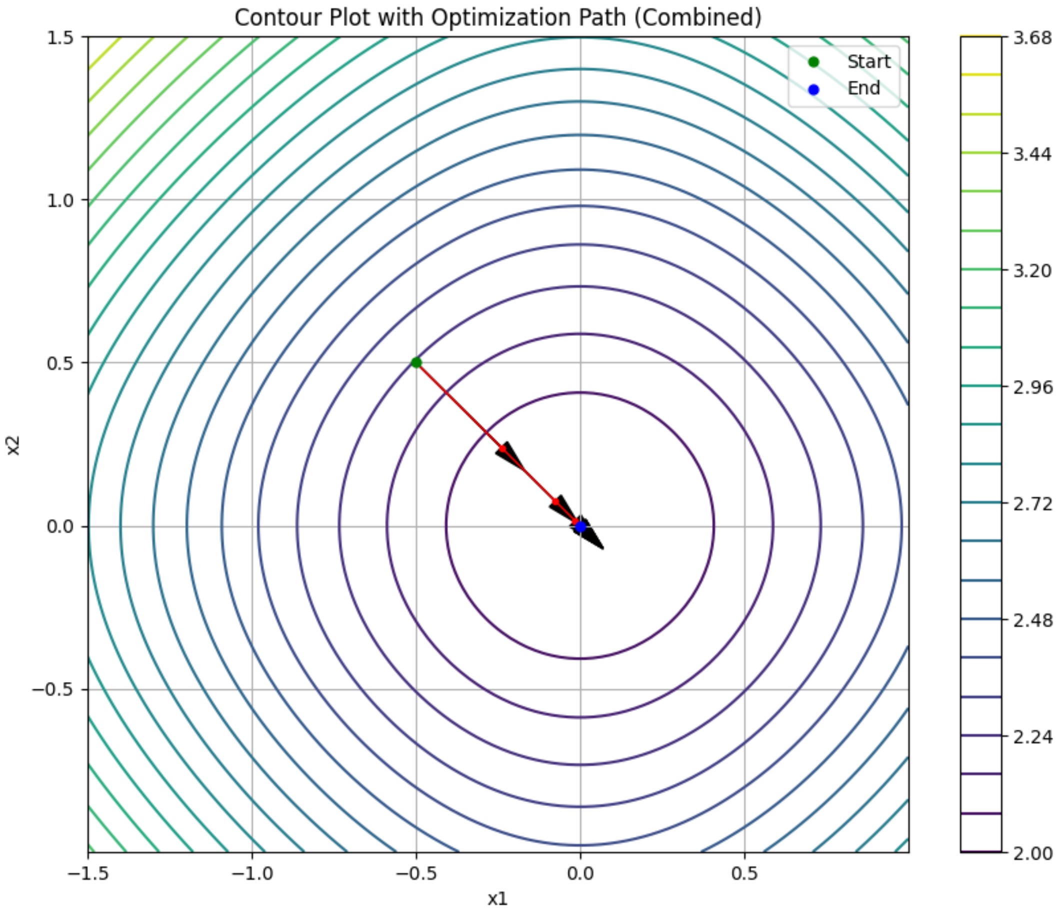

Contour Plot with Optimization Path (Pure)
Start
End
x1
x2
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.00
2.24
2.48
2.72
2.96
3.20
3.44
3.68

PROF


Contour Plot with Optimization Path (Damped)
Start
End
x1
x2
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.00
2.24
2.48
2.72
2.96
3.20
3.44
3.68


Contour Plot with Optimization Path (Levenberg-Marquardt)
Start
End
x1
x2
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
3.68
3.44
3.20
2.96
2.72
2.48
2.24
2.00

PROF



6. Final Results

Test case	Backtracking-Armijo	Backtracking-Goldstein	Bisection	Pure	Damped	Levenberg-Marquardt	Combined
0	[2. 2.]	[2. 2.]	[2. 2.]	[2. 2.]	[2. 2.]	[2. 2.]	[2. 2.]
1	[2. 2.]	[2. 2.]	[2. 2.]	[2. 2.]	[2. 2.]	[2. 2.]	[2. 2.]
2	[-1.748 0.874]	[-1.753 0.953]	[-1.748 0.874]	[-1.748 0.874]	[-1.748 0.874]	[-1.748 0.874]	[-1.748 0.874]
3	[1.748 -0.874]	[1.753 -0.953]	[1.748 -0.874]	[1.748 -0.874]	[1.748 -0.874]	[1.748 -0.874]	[1.748 -0.874]
4	[-0. 0.]	[-0. 0.]	[-0. 0.]	[-1.748 0.874]	[-1.748 0.874]	[-1.748 0.874]	[-1.748 0.874]
5	[0. -0.]	[0. -0.]	[0. -0.]	[1.748 -0.874]	[1.748 -0.874]	[1.748 -0.874]	[1.748 -0.874]
6	[0.993 0.986 0.972 0.945]	[1.056 0.794 0.135 -1.148]	[0.998 0.996 0.992 0.984]	[1. 1. 1. 1.]	[1. 1. 1. 1.]	[1. 1. 1. 1.]	[1. 1. 1. 1.]

PROF

Test case	Backtracking-Armijo	Backtracking-Goldstein	Bisection	Pure	Damped	Levenberg-Marquardt	Combined
7	[0.991 0.981 0.963 0.926]	[-0.456 0.156 -1.014 1.533]	[0.998 0.996 0.992 0.984]	[1. 1. 1. 1.]	[1. 1. 1. 1.]	[1. 1. 1. 1.]	[1. 1. 1. 1.]
8	[-0.861 0.753 0.572 0.327]	[-0.537 0.748 1.35 2.057]	[-0.777 0.616 0.385 0.148]	[-0.776 0.613 0.382 0.146]	[-0.776 0.613 0.382 0.146]	[-0.776 0.613 0.382 0.146]	[-0.776 0.613 0.382 0.146]
9	[-0.817 0.679 0.467 0.218]	[-0.518 0.068 0.068 3.586]	[0.998 0.996 0.992 0.984]	[1. 1. 1. 1.]	[1. 1. 1. 1.]	[1. 1. 1. 1.]	[1. 1. 1. 1.]
10	[-2.904 -2.904 -2.904 -2.904]	[0. 0. 0. 0.]	[-2.904 -2.904 -2.904 -2.904]	[0.157 0.157 0.157 0.157]	[0. 0. 0. 0.]	[0.157 0.157 0.157 0.157]	[-2.904 -2.904 -2.904 -2.904]
11	[2.747 2.747 2.747 2.747]	[2.747 2.747 2.747 2.747]	[2.747 2.747 2.747 2.747]	[2.747 2.747 2.747 2.747]	[2.747 2.747 2.747 2.747]	[2.747 2.747 2.747 2.747]	[2.747 2.747 2.747 2.747]
12	[-2.904 -2.904 -2.904 -2.904]	[-2.904 -2.904 -2.904 -2.904]	[-2.904 -2.904 -2.904 -2.904]	[-2.904 -2.904 -2.904 -2.904]	[-2.904 -2.904 -2.904 -2.904]	[-2.904 -2.904 -2.904 -2.904]	[-2.904 -2.904 -2.904 -2.904]
13	[2.747 -2.904 2.747 -2.904]	[2.747 -2.904 2.747 -2.904]	[2.747 -2.904 2.747 -2.904]	[2.747 -2.904 2.747 -2.904]	[2.747 -2.904 2.747 -2.904]	[2.747 -2.904 2.747 -2.904]	[2.747 -2.904 2.747 -2.904]
14	[0. 0.]	[3. 3.]	[-0. -0.]	[-8.71896425e+115 -8.71896425e+115]	[-0. -0.]	[0. 0.]	[0. 0.]
15	[-0. 0.]	[-0. 0.]	[-0. 0.]	[0. -0.]	[0. -0.]	[-0. 0.]	[-0. 0.]
16	[-0. 0.]	[-3.5 0.5]	[0. 0.]	[1.61634765e+132 0.00000000e+000]	[-0. 0.]	[0. 0.]	[0. 0.]

PROF

7. Conclusion

In this assignment, we implemented and analyzed various optimization algorithms including Steepest Descent with different line search methods and Newton's Method with various modifications.

The key findings are:

1. **Convergence Performance**:
    - Newton's methods typically converged faster than steepest descent methods in terms of iteration count

    - The combined approach of Damped Newton with Levenberg-Marquardt provided the most robust convergence across different functions

2. **Line Search Methods**:
    - Bisection with Wolfe conditions generally provided more reliable step sizes compared to backtracking methods
    - Backtracking-Goldstein was more stable than Backtracking-Armijo alone

3. **Function-Specific Observations**:
    - Rosenbrock function was the most challenging to optimize due to its narrow valley structure
    - Three Hump Camel function demonstrated the value of robust optimization methods for multimodal functions
    - Styblinski-Tang function showed how the algorithms handle higher-dimensional optimization

4. **Practical Considerations**:
    - The choice of initial point significantly affected convergence behavior
    - Newton's methods required more computation per iteration but fewer iterations overall
    - For ill-conditioned problems, Levenberg-Marquardt modifications were essential for reliable convergence

This report demonstrates the importance of understanding the theoretical foundations of optimization algorithms and their practical implementation considerations.