main

August 29, 2024

David Torres Treviño A00824753

Luis Kevin Cepeda Zapata A00824840

[1]:
```python
from functions import *
```

0.0.1 G. Explain which method you select to solve each of the problems and explain why.

All of the three problems were solved using the Hill Climber, Gradient descent with wolfe conditions and Newton Method. In general, the fastest method was the Newton Method and achieved more exact solutions, when there was no local minima. If the problem had local minima all of the methods got stuck in there (you could modify the ball radius of hill climber to get around this by seeing the graph and choosing a more optimal value, but this wouldn't be possible for black boxes problem).

0.0.2 H. Can evolutionary algorithms help to solve any of the previous problems? Why?

Evolutionary algorithms, such as Differential Evolution (DE), can help solve many of the previous optimization problems.

1. Global Search Capability: Differential Evolution is designed to explore the entire solution space, making it effective for finding global minima in functions that have multiple local minima.
2. No Need for Gradient Information: DE does not rely on gradient information, which is particularly useful when dealing with functions that are non-differentiable, noisy, or complex.

1 1. Classical optimization methods

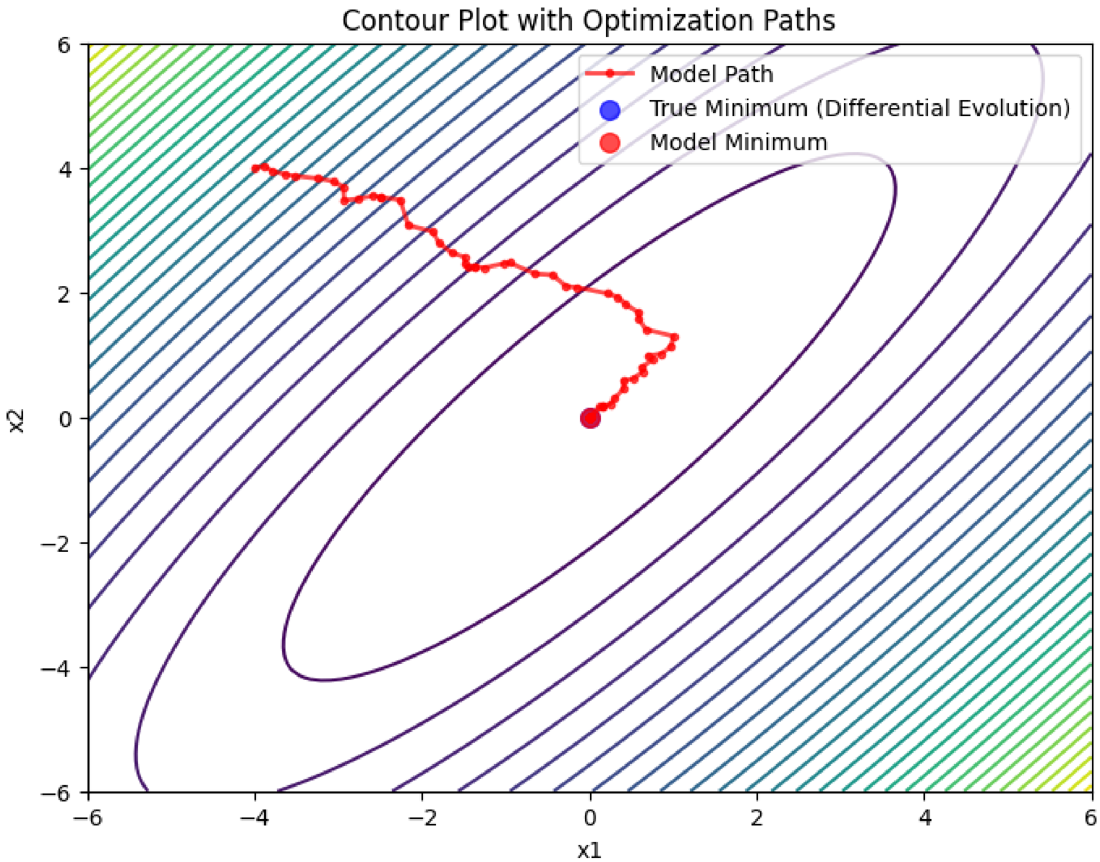
1.0.1 F_A

Algorithm	Point Found	Evaluation	Iterations	Real Minimum	Two Norm Error
Hill Climber	[-0.0041513, -0.0100955]	1.3000616	600	1.30	0.0109156
Gradient Descent	[0.0002476, 0.0003736]	1.3000001	48	1.30	0.0004481
Newton Method	[0.0000000, 0.0000000]	1.3000000	1	1.30	8.6140e-08

Hill Climber

[2]:
```
x_init = np.array([-4,4])
constraints = [[-6, 6],[-6, 6]]
x_best, fx_best, x_history, fx_history = hill_climber(f_A, delta=0.1,␣
↪n_iter=600, n_intervals=100, x_init=x_init, constraints = constraints)
print()
plot_function_with_paths(f_A, constraints, x_history, fx_history)
```

```
i = 600, x1 = -0.0041513, x2 = -0.0100955, fx_best = 1.3000616
```



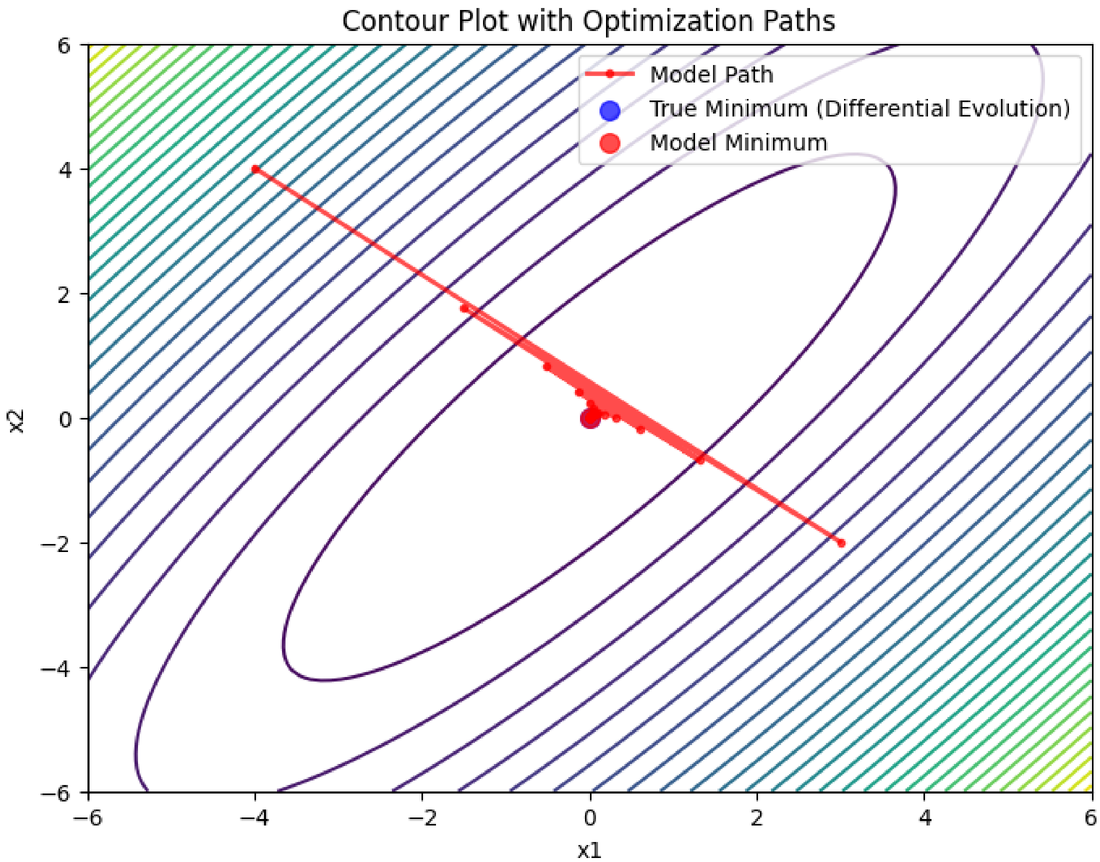

```
True minimum found by differential evolution at x1 = -0.00, x2 = -0.00, with
value = 1.30
Model minimum at x1 = -0.00, x2 = -0.01, with value = 1.30
L2 norm error for parameters: 0.010915652409520997
```

Gradient Descent with Wolfe Conditions

[3]:
```
x_best, fx_best, x_history, fx_history, i = grad_descent(f_A, tol=0.0005,␣
↪max_iter=100, n_intervals=8, x_init=x_init, constraints = constraints)
print()
```

```
plot_function_with_paths(f_A, constraints, x_history, fx_history)
```

i = 48, x1 = 0.0002476, x2 = 0.0003736, fx_best = 1.30000010

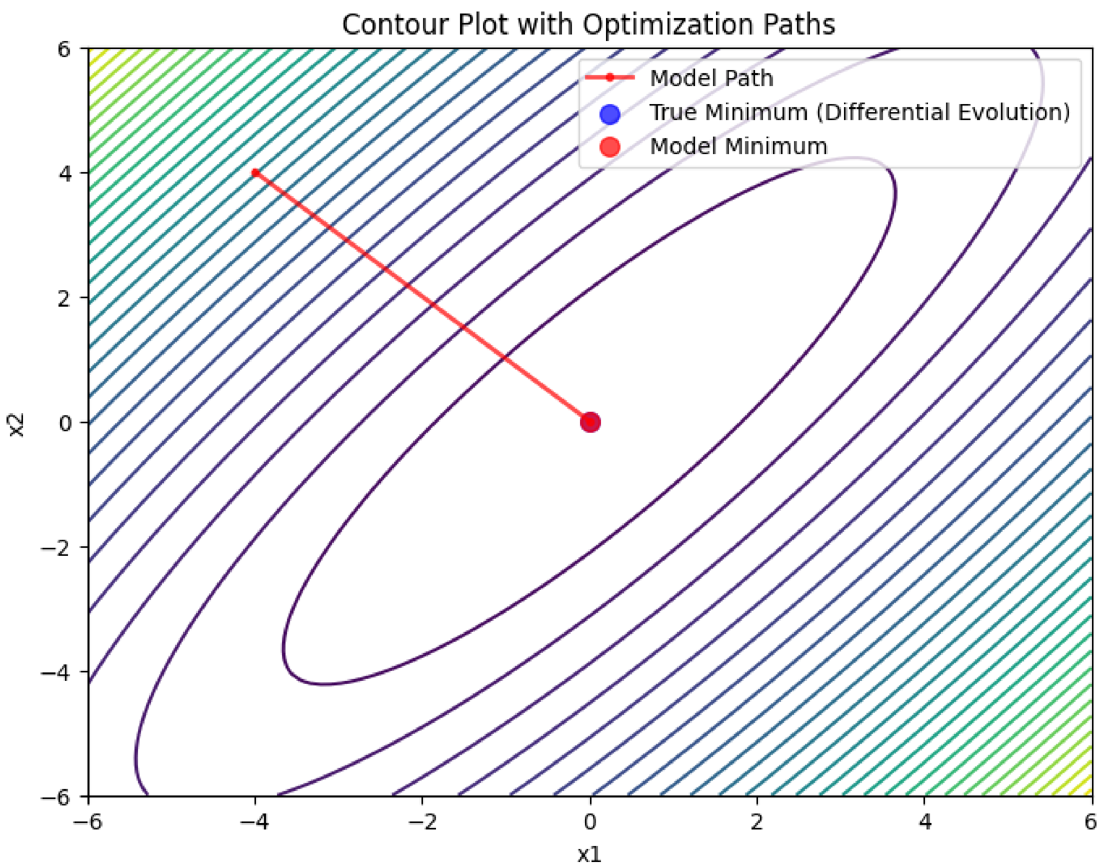


True minimum found by differential evolution at x1 = 0.00, x2 = 0.00, with value = 1.30
Model minimum at x1 = 0.00, x2 = 0.00, with value = 1.30
L2 norm error for parameters: 0.0004481906623516091

Newton Method

[4]:
```
x_best, fx_best, x_history, fx_history, i = newton_method(f_A, tol=0.0005,
    max_iter=100, n_intervals=6, x_init=x_init, constraints = constraints)
print()
plot_function_with_paths(f_A, constraints, x_history, fx_history)
```

i = 1, x1 = 0.0000000, x2 = 0.0000000, fx_best = 1.3000000




```
True minimum found by differential evolution at x1 = 0.00, x2 = 0.00, with value
= 1.30
Model minimum at x1 = 0.00, x2 = 0.00, with value = 1.30
L2 norm error for parameters: 8.614083252913067e-08
```

1.0.2 F_B

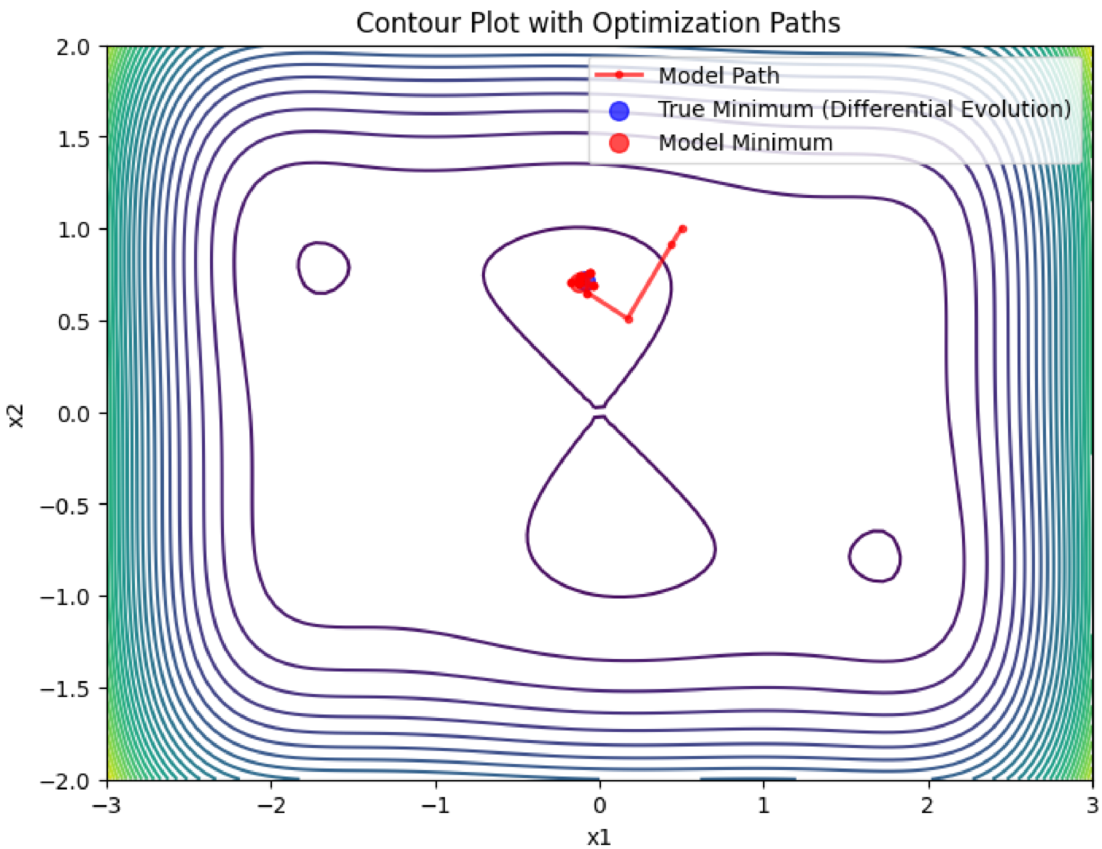
Algorithm	Point Found	Evaluation	Iterations	Real Minimum	Two Norm Error
Hill Climber	[-0.1235228, 0.7081489]	-1.0269200	300	-1.03	0.0339810
Gradient Descent	[0.0898398, -0.7126366]	-1.03162855	8	-1.03	1.9900e-05
Newton Method	[-0.4096386, -0.8975904]	0.3550784	50	-1.03	0.5326174

Hill Climber

[11]:
```
x_init=np.array([0.5,1])
constraints = [[-3, 3],[-2, 2]]
```

```
x_best, fx_best, x_history, fx_history = hill_climber(f_B, delta=0.2,␣
 ↪n_iter=300, n_intervals=50, x_init=x_init, constraints = constraints)
print()
plot_function_with_paths(f_B, constraints, x_history, fx_history)
```

```
i = 300, x1 = -0.1235228, x2 = 0.7081489, fx_best = -1.0269200
```

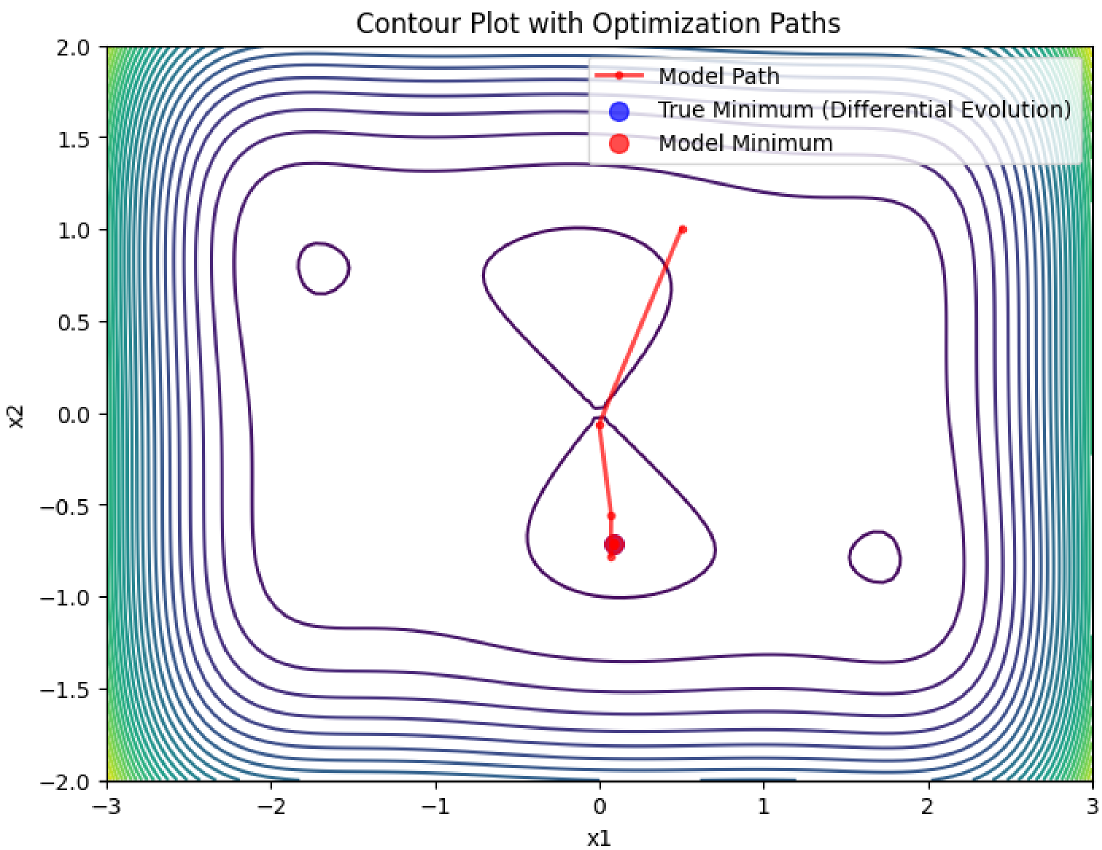



```
True minimum found by differential evolution at x1 = -0.09, x2 = 0.71, with
value = -1.03
Model minimum at x1 = -0.12, x2 = 0.71, with value = -1.03
L2 norm error for parameters: 0.03398104693751721
```

Gradient Descent with Wolfe Conditions

[19]:
```
x_best, fx_best, x_history, fx_history, i = grad_descent(f_B, tol=0.0005,␣
 ↪max_iter=100, n_intervals=1, x_init=x_init, constraints = constraints)
print()
plot_function_with_paths(f_B, constraints, x_history, fx_history)
```

```
i = 8, x1 = 0.0898398, x2 = -0.7126366, fx_best = -1.03162855
```

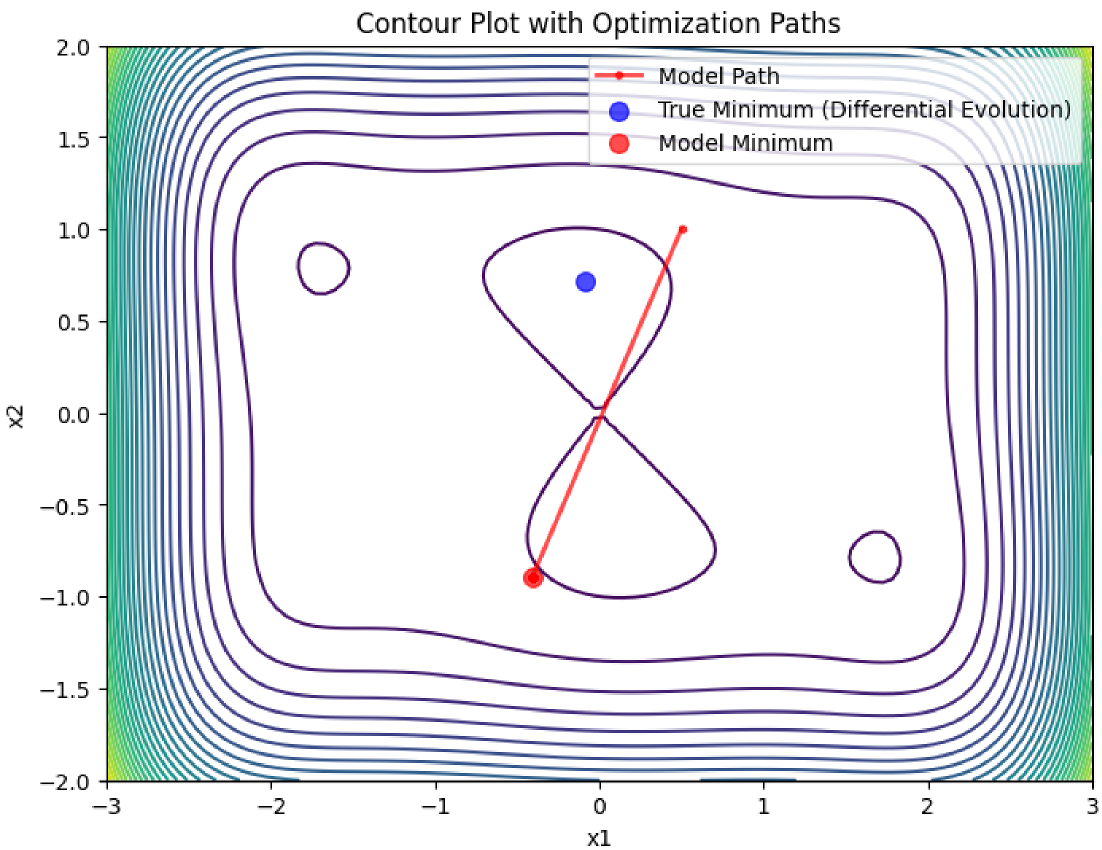



```
True minimum found by differential evolution at x1 = 0.09, x2 = -0.71, with
value = -1.03
Model minimum at x1 = 0.09, x2 = -0.71, with value = -1.03
L2 norm error for parameters: 1.990040820361988e-05
```

Newton Method

[20]:
```python
x_best, fx_best, x_history, fx_history, i = newton_method(f_B, tol=0.0008,
    max_iter=50, n_intervals=5, x_init=x_init, constraints = constraints)
print()
plot_function_with_paths(f_B, constraints, x_history, fx_history)
```

```
i = 50, x1 = -0.4096386, x2 = -0.8975904, fx_best = 0.3550784
```




```
True minimum found by differential evolution at x1 = -0.09, x2 = 0.71, with
value = -1.03
Model minimum at x1 = -0.41, x2 = -0.90, with value = 0.36
L2 norm error for parameters: 1.6416955475897947
```

1.0.3 F_C

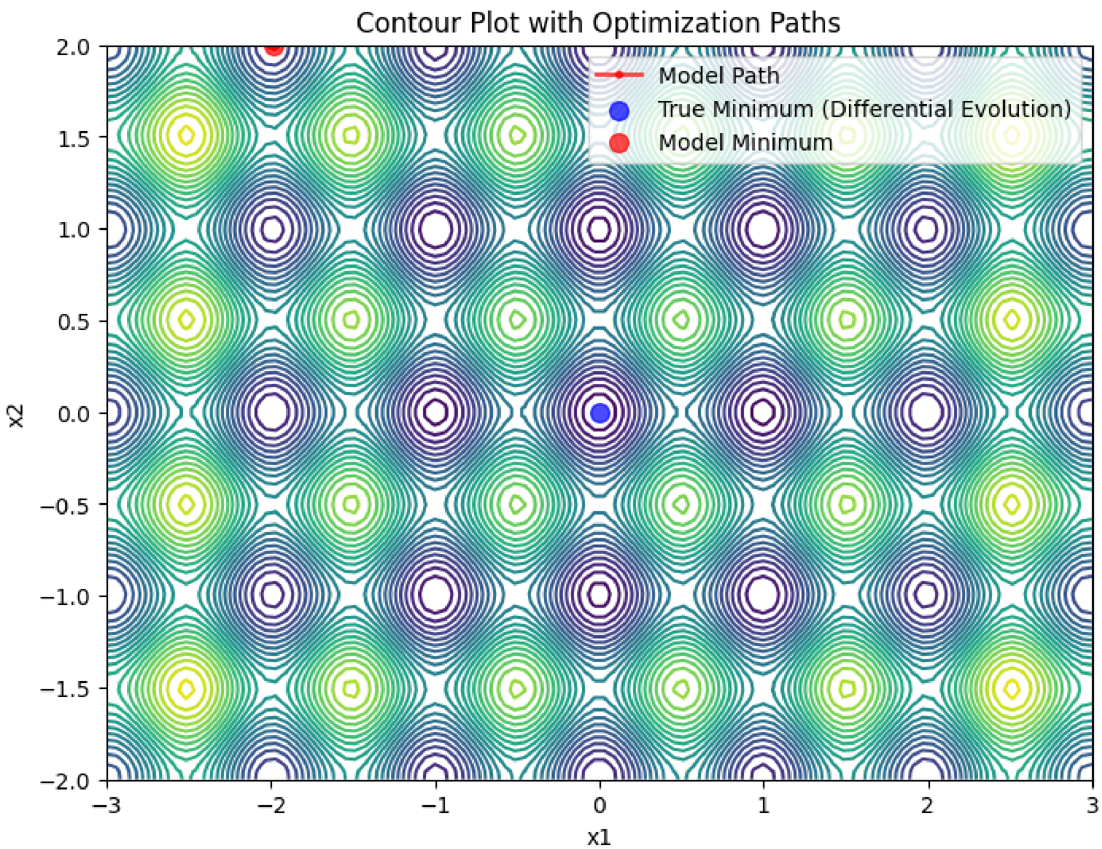
Algorithm	Point Found	Evaluation	Iterations	Real Minimum	Two Norm Error
Hill Climber	[-1.9861292, 2.0028127]	7.9954832	500	0	2.8206325
Gradient Descent	[-2, 2]	8	500	0	2.8284271
Newton Method	[-2, 2]	8	150	0	2.8284271

Hill Climber

[8]:
```
x_init=np.array([-2,2])
x_best, fx_best, x_history, fx_history = hill_climber(f_C, delta=0.2,␣
↪n_iter=500, x_init=x_init, constraints = constraints)
```

```
print()
plot_function_with_paths(f_C, constraints, x_history, fx_history)
```

```
i = 500, x1 = -1.9827902, x2 = 1.9968675, fx_best = 7.9792798
```

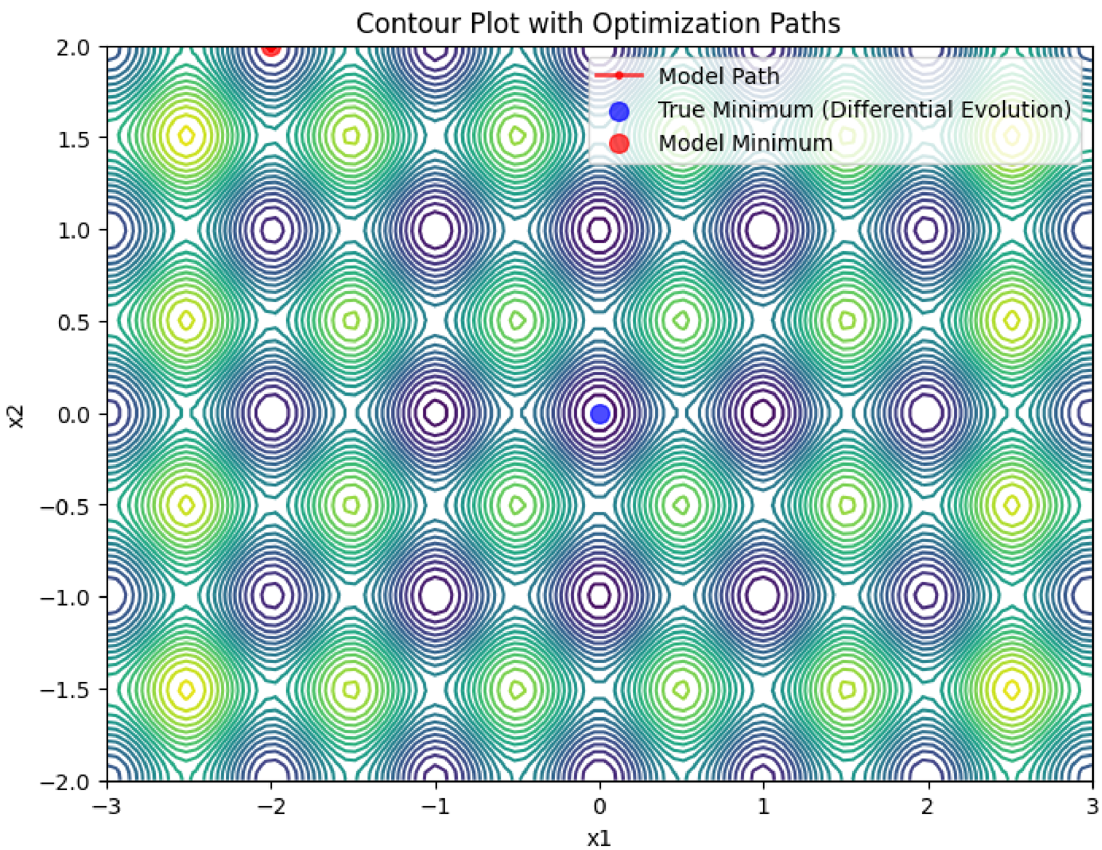



```
True minimum found by differential evolution at x1 = 0.00, x2 = -0.00, with
value = 0.00
Model minimum at x1 = -1.98, x2 = 2.00, with value = 7.98
L2 norm error for parameters: 2.814060535192687
```

Gradient Descent with Wolfe Conditions

[9]:
```
x_init=np.array([-2,2])
x_best, fx_best, x_history, fx_history, i = grad_descent(f_C, tol=0.0005,
    max_iter=500, x_init=x_init,  constraints = constraints)
print()
plot_function_with_paths(f_C, constraints, x_history, fx_history)
```

```
i = 500, x1 = -2.0000000, x2 = 2.0000000, fx_best = 8.0000000
```

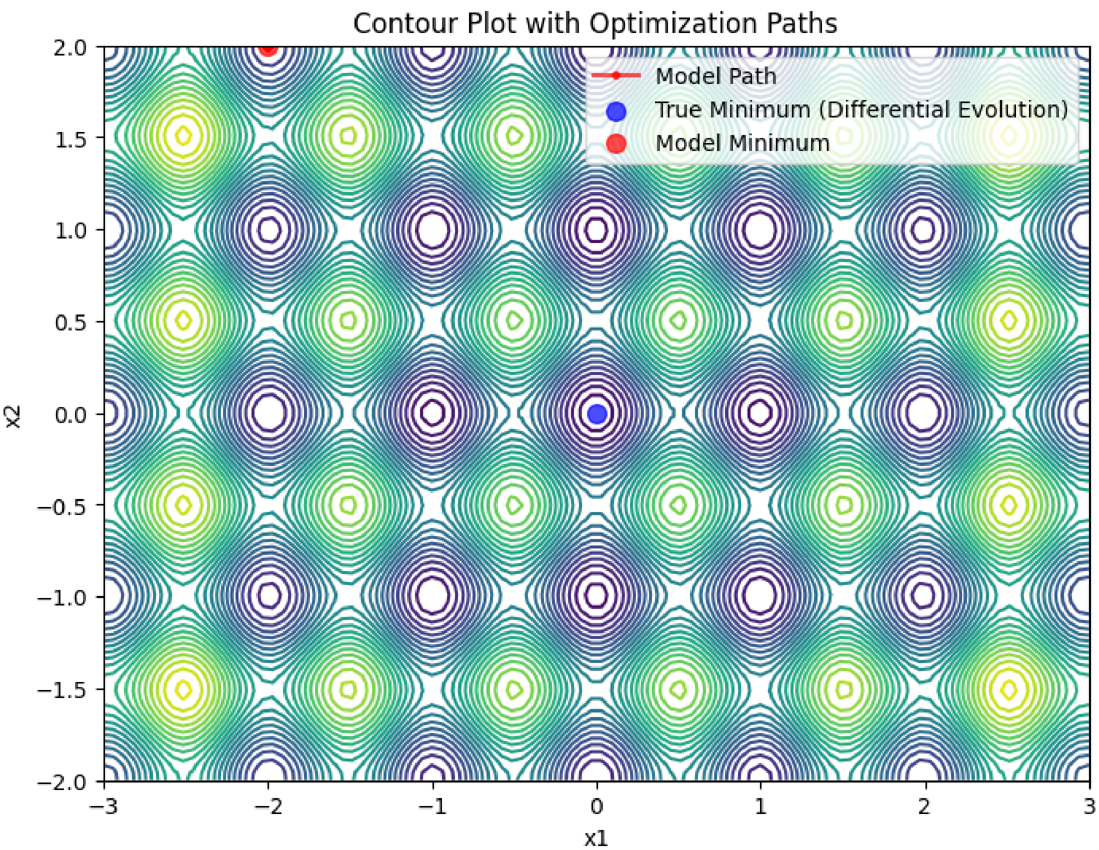



```
True minimum found by differential evolution at x1 = 0.00, x2 = -0.00, with
value = 0.00
Model minimum at x1 = -2.00, x2 = 2.00, with value = 8.00
L2 norm error for parameters: 2.8284271269094416
```

Newton Method

[10]:
```
x_init=np.array([-2,2])
x_best, fx_best, x_history, fx_history, i = newton_method(f_C, tol=0.0005,
    max_iter=150, x_init=x_init, constraints = constraints)
print()
plot_function_with_paths(f_C, constraints, x_history, fx_history)
```

```
i = 150, x1 = -2.0000000, x2 = 2.0000000, fx_best = 8.0000000
```




```
True minimum found by differential evolution at x1 = 0.00, x2 = 0.00, with value
= 0.00
Model minimum at x1 = -2.00, x2 = 2.00, with value = 8.00
L2 norm error for parameters: 2.8284271235955014
```

[24]:
```
!choco install pandoc
```

```
/bin/bash: line 1: choco: command not found
```

[ ]:
```

```