SFU SCHOOL OF MECHATRONIC SYSTEMS ENGINEERING

MSE 211: Computational Methods
Lab 2 Report

Group 3
David Go | 301460356
Sarah Li | 301420565
Kenneth Tubungbanua | 301444650

Table of Contents

Introduction **3**
Results and Discussion **3**
Part I: Comparison of Optimization **3**
Steepest Descent Method 3
Newton's Method 7
Matlab's Function 8
Part II: Synthesis of Mechanisms **10**
Motion Generation 10
Path Generation 11
Function Generation 12
Conclusion **13**
Appendix **14**

Introduction

The goal of this lab is to solve two optimization problems using Matlab. The first problem is unconstrained where three different algorithms are used to minimize the function. These methods are the steepest descent method, Newton's method, and matlab's function that is given in the code. In addition, the effect of the initial guesses is also observed in this lab. The second problem is in relation to the synthesis or design of mechanisms for motion, function and path generation, using optimization to solve this problem.

Results and Discussion

Part I: Comparison of Optimization

For this lab, it is Himmelblau's function that is minimized in order to determine its local minima. Three different methods are used to solve this problem which are steepest descent method, Newton's method, and a matlab function that is predetermined for this experiment. Figure 1 below shows a picture of Himmelblau's function which will be minimized to determine the local minima. Taking a look at the function, it should have at least 4 local minima due to two polynomials having both a degree of two. Further analysis of Himmelblau's function using various methods are discussed below in more details.

Figure 1: Himmelblau's function

$$f(x,y) = (x^2 + y - 11)^2 + (x + y^2 - 7)^2$$

Steepest Descent Method

To use the steepest descent method, a new **x** is calculated by taking the difference of the current value and product of the gradient evaluated at the current value of **x** and the step size. To calculate the gradient, the first partial derivatives of Himmelblau's function with respect to both x and y are put into a vector on Matlab. To calculate the step size, the Hessian needs to be determined first which is found by taking the second partial derivatives of Himmelblau's function. Formulas for the gradient, step size, and the Hessian respectively are shown in Figure 1 of the appendix. These formulas would correspond to a separate function that was created on Matlab that the main function would call upon to perform the calculations as **x** changes after each iteration. The results of the steepest descent method for the initial guesses are shown in the figures below.

Figure 2: Table showing location of local minima for Point 1

Point 1(0,0)

k	x	y	f	ea
1	0.457341	0.718678	137.7631	0.851856
2	1.923282	2.52082	24.4828	2.32308
3	5.534742	2.169977	485.9063	3.628461
4	4.083994	1.957164	59.14735	1.466274
5	3.340897	1.825471	4.055198	0.754676
6	3.085708	1.809823	0.517943	0.255668
7	2.974124	1.96698	0.059802	0.192742
8	3.006538	1.987474	0.0026	0.038349
9	2.99872	1.997183	0.000267	0.012466
10	3.000662	1.998742	2.65E-05	0.002491
11	2.999873	1.999716	2.68E-06	0.001253
12	3.000067	1.999873	2.70E-07	0.000249
13	2.999987	1.999971	2.73E-08	0.000127
14	3.000007	1.999987	2.76E-09	2.51E-05

Figure 3: Table showing location of local minima for Point 2

Point 2 (-1,0)				
k	x	y	f	ea
1	-1.92075	0.767296	112.2386	1.198554
2	-5.97388	5.429452	1179.431	6.177662
3	-4.17489	4.3325	173.5186	2.107053
4	-3.27559	3.495379	14.17157	1.228617
5	-2.90237	3.168516	0.36955	0.496123

6	-2.81168	3.128367	0.001773	0.099179
7	-2.80552	3.131863	1.73E-05	0.007075
8	-2.80518	3.131282	1.79E-07	0.000673
9	-2.80512	3.131318	1.85E-09	7.11E-05

Figure 4: Table showing location of local minima for Point 3

Point 3(1,-5)				
k	x	y	f	ea
1	1.079899	-3.51097	219.13	1.491177
2	1.536992	-2.32144	120.1073	1.274323
3	5.160064	-1.18222	208.8257	3.797955
4	4.072288	-1.2956	19.94675	1.093667
5	3.622769	-1.39604	2.570609	0.460605
6	3.494024	-1.55378	1.311319	0.203609
7	3.620149	-1.66338	0.57126	0.167089
8	3.550946	-1.72973	0.223442	0.095877
9	3.601593	-1.78608	0.07777	0.075763
10	3.573875	-1.80894	0.024995	0.035929
11	3.590323	-1.82962	0.007559	0.026427
12	3.581364	-1.83652	0.002207	0.011306
13	3.586183	-1.84287	0.00063	0.007968
14	3.583565	-1.84484	0.000178	0.003276
15	3.584926	-1.84665	4.99E-05	0.002272
16	3.584187	-1.84721	1.39E-05	0.000922

17	3.584568	-1.84772	3.88E-06	0.000637
18	3.584361	-1.84787	1.08E-06	0.000258
19	3.584467	-1.84801	3.01E-07	0.000178
20	3.58441	-1.84806	8.39E-08	7.17E-05

Figure 5: Table showing location of local minima for Point 4

Point 4 (-5,0)				
k	x	y	f	ea
1	-3.82669	-0.10807	129.4641	1.178277
2	-3.26125	-0.19575	104.8219	0.5722
3	-3.07253	-0.29462	103.1527	0.213048
4	-3.15548	-0.532	99.94591	0.25145
5	-3.15925	-1.0359	86.78133	0.503923
6	-3.43706	-2.23093	31.82129	1.226889
7	-4.04123	-5.46661	355.0614	3.291603
8	-4.16504	-4.18628	45.11985	1.286296
9	-4.0121	-3.51525	4.310475	0.688243
10	-3.78256	-3.35597	0.232736	0.279389
11	-3.79803	-3.29611	0.021009	0.061832
12	-3.77935	-3.2898	0.001928	0.019707
13	-3.78105	-3.28435	0.000179	0.005708
14	-3.77931	-3.2838	1.66E-05	0.001824
15	-3.77947	-3.28329	1.55E-06	0.000531
16	-3.77931	-3.28324	1.44E-07	0.00017

17	-3.77933	-3.2832	1.34E-08	4.95E-05

From the Matlab results, the number of local minima is four which agrees with the initial predictions. The locations of the local minima would be **x = 3, x = -2.0512, x = 3.58841,** and **x = -3.77931**. A plot of the path followed and contours for each of these points are shown in Figure 2 of the appendix.

Newton's Method

Newton's Method differs from the Steepest Descent Method as the difference between the current value of **x** and the quotient of the first derivative and the second derivative of the function. The results of Newton's method evaluated at the four points are shown in the figures below.

Figure 6: Table showing location of local minima for Point 1

Point 1(0,0)				
k	x	y	f	ea
1	-0.33333	-0.84615	181.5006	0.909443
2	-0.27076	-0.91915	181.6164	0.096146
3	-0.27085	-0.92303	181.6165	0.003878
4	-0.27084	-0.92304	181.6165	1.04E-05

Figure 7: Table showing location of local minima for Point 2

Point 2 (-1,0)				
k	x	y	f	ea
1	-0.09502	-0.78733	180.6557	1.19953
2	-0.27265	-0.92129	181.6164	0.222481
3	-0.27084	-0.92304	181.6165	0.002514
4	-0.27084	-0.92304	181.6165	2.33E-06

Figure 8: Table showing location of local minima for Point 3

Point 3(1,-5)				
k	x	y	f	ea
1	0.104549	-3.57672	246.942	1.681537
2	-0.0102	-2.70195	187.8248	0.882263
3	-0.08773	-2.21953	179.2243	0.488609
4	-0.12003	-2.00783	178.3688	0.214147
5	-0.12751	-1.95681	178.3373	0.05157
6	-0.12796	-1.95373	178.3372	0.003113
7	-0.12796	-1.95371	178.3372	1.12E-05

Figure 9: Table showing location of local minima for Point 4

Point 4 (-5,0)				
k	x	y	f	ea
1	-3.81448	0.093251	130.0399	1.189185
2	-3.24797	-0.03252	105.2327	0.580298
3	-3.08663	-0.07743	104.0221	0.167471
4	-3.07312	-0.08133	104.0152	0.014066
5	-3.07303	-0.08135	104.0152	9.67E-05

From the results of Newton's method, the location of the four points respectively are **x = -0.27084, x = -0.27084, x = -0.12796,** and **x = -3.07303**. When compared to the Steepest Descent method, it took less iterations until the value of x converged to one value. Images of the path followed and contours are shown in Figure 3 of the appendix.

Matlab's Function

A given function in matlab was used to determine the number of iterations, and the locations for each of the maxima as shown in the figures below.

Figure 10: Table showing number of iterations and the location of local minima for Point 1

Point 1(0,0)	
Number of Iterations	129
final estimate point	(3,2)
function evaluation	1.6735E-13

Figure 11: Table showing number of iterations and the location of local minima for Point 2

Point 2 (-1,0)	
Number of Iterations	8
final estimate point	(-2.8051,3.1313)
function evaluation	1.0037E-13

Figure 12: Table showing number of iterations and the location of local minima for Point 3

Point 3(1,-5)	
Number of Iterations	14
final estimate point	(3.5844,-1.8481)
function evaluation	1.169E-010

Figure 13: Table showing number of iterations and the location of local minima for Point 4

Point 4 (-5,0)	
Number of Iterations	12
final estimate point	(-3.779,-3.2832)
function evaluation	5.4331E-12

Upon further observations of the results, it would be Point 2 that converged to the local minima at a faster rate. On the other hand, Point 1 would take more iterations until it reached the minimum. From this, the values of the initial points would affect the number of iterations it would take to find the local minima. It is important to note that on average it would be Newton's Method that is the quickest to determine the values of the local minima when compared to the Steepest Descent method or the given function on Matlab.

Part II: Synthesis of Mechanisms

In this part, a mechanism will be designed to accomplish a particular task for motion generation, function generation and path generation. All three tasks will require the mechanism to pass through four precision points, thus the goal is to find the length of all the links, as well as the location of the ground pivots. It is important to note that while running the functions for motion, function, and path generation the solver stopped prematurely. The last iteration all three methods stopped at was the 700th iteration.

Part II.1 Motion Generation

Motion Generation

For motion generation, the mechanism was divided into the left side and right side. The precision points (**δ**) has been defined and were represented as vectors. These vectors represent the relative motion of a single point of the box that undergoes rotational motion. After determining the four precision points, their x and y components can be determined using the function below. The x and y components would be the input of another function called **F** for the left side and **G** for the right side. The values for both **F** and **G** are shown below.

Figure 14a): Vector representations for the four precision points for both left and right side respectively

$$f_{jx} = r_{2x} + r'_{3x} + \delta_{jx} - r'_{3x}\cos(\alpha_j) + r'_{3y}\sin(\alpha_j) - r_{2x}\cos(\beta_j) + r_{2y}\sin(\beta_j) = 0$$

$$f_{jy} = r_{2y} + r'_{3y} + \delta_{jy} - r'_{3y}\cos(\alpha_j) - r'_{3x}\sin(\alpha_j) - r_{2y}\cos(\beta_j) - r_{2x}\sin(\beta_j) = 0$$

$$g_{jx} = r_{4x} + r''_{3x} + \delta_{jx} - r''_{3x}\cos(\alpha_j) + r''_{3y}\sin(\alpha_j) - r_{4x}\cos(\gamma_j) + r_{4y}\sin(\gamma_j) = 0$$

$$g_{jy} = r_{4y} + r''_{3y} + \delta_{jy} - r''_{3y}\cos(\alpha_j) - r''_{3x}\sin(\alpha_j) - r_{4y}\cos(\gamma_j) - r_{4x}\sin(\gamma_j) = 0$$

To verify that the mechanism will travel along the desired path, the vector representations in figure 14 can be used in an objective function. When this function reaches zero, the mechanism's trajectory takes it through all four precision points. Part II.2 and Part II.3 use functions this way as well.

Figure 14b): Vector representations used in an objective function

$$F = \sqrt{f_{1x}^2 + f_{2x}^2 + f_{3x}^2 + f_{1y}^2 + f_{2y}^2 + f_{3y}^2}$$

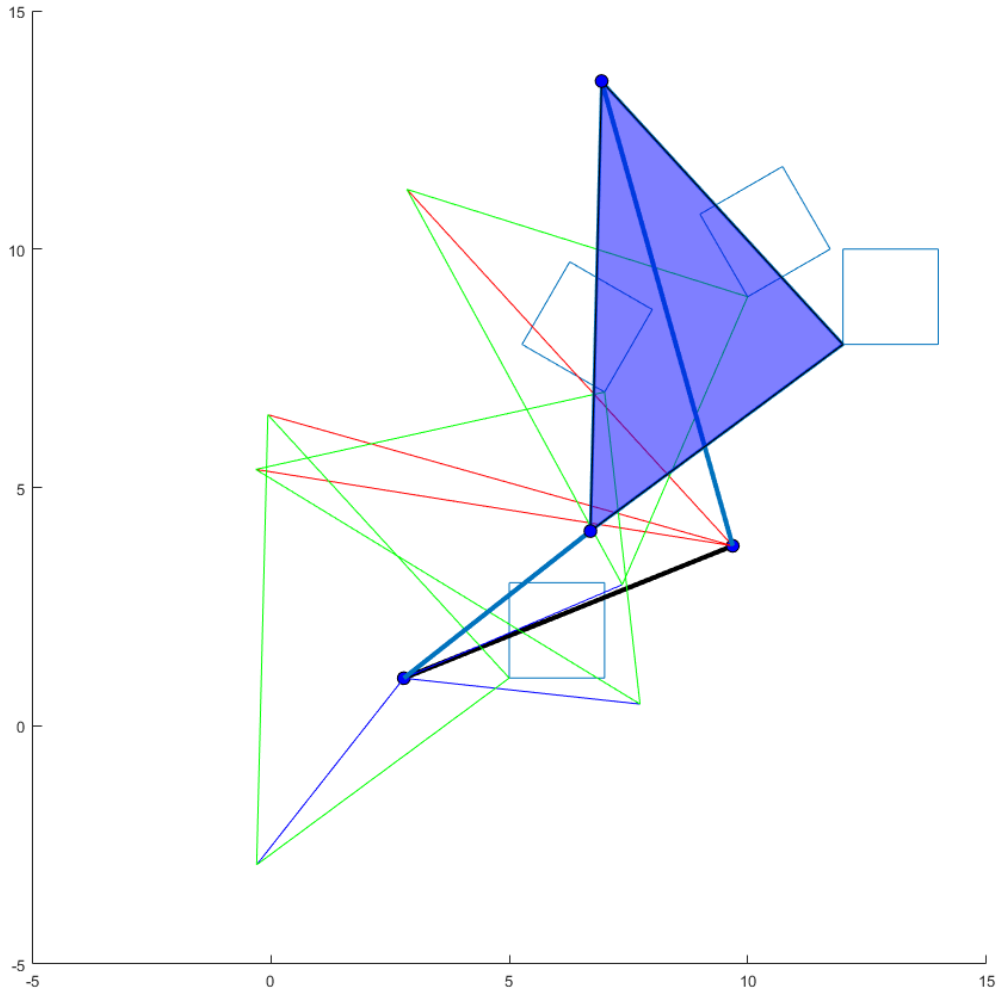
Figure 15: Values of the objective function F and G

```
F = 4.2148e-05
G = 9.5614e-04
```

Figure 16: Initial conditions for motion generation

Initial Conditions										
	R_2	R'_3	β_1	β_2	β_3	R_4	R''_3	γ_1	γ_2	γ_3
x (θ)	-3	2	(-30)	(-100)	(-250)	-3	-2	(-40)	(-90)	(-200)
y	1	4				3	4			

Figure 17: Final frame of the motion generation render



Path Generation

The path generation is solved in a similar way as the motion generation by determining the values of the x and y components for the four precision points. They are then evaluated through the objective function **G** as shown below.

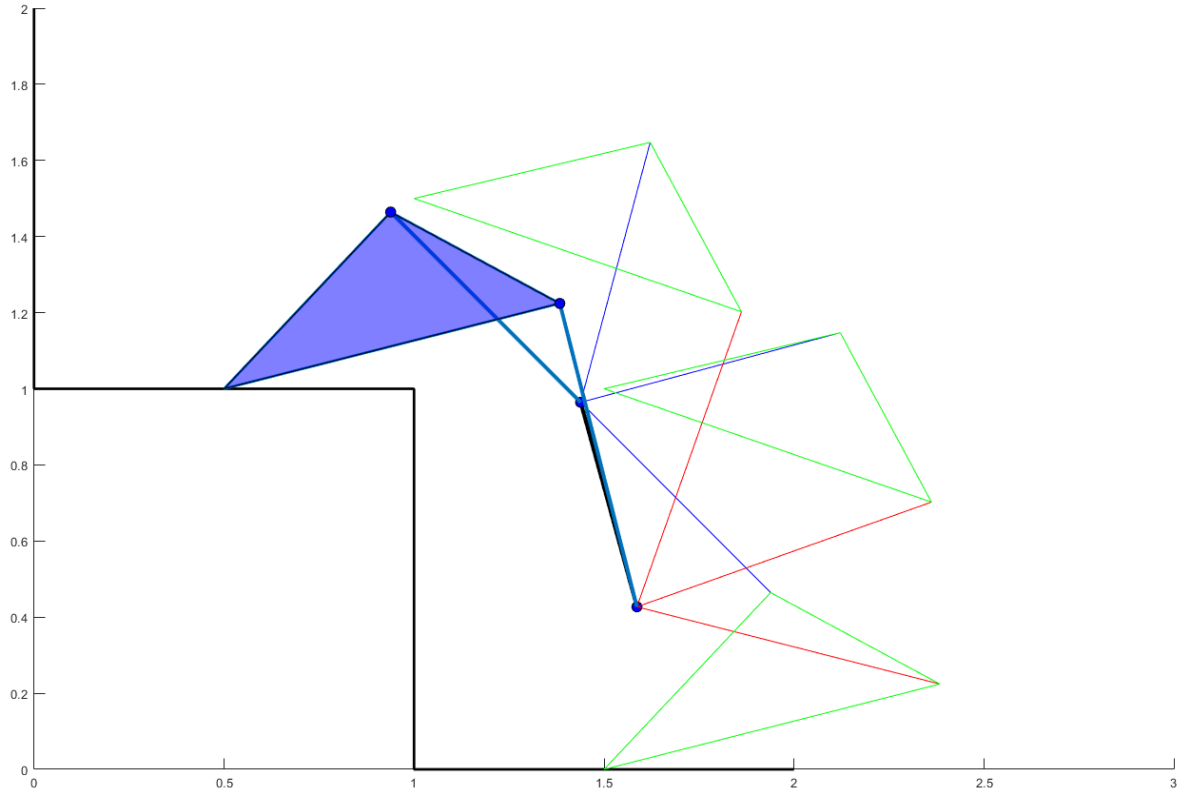
Figure 18: Initial conditions for path generation

Initial Conditions										
	R_2	R'_3	α_1	α_2	α_3	R_4	R''_3	γ_1	γ_2	γ_3
x (θ)	-7	0.5	(-30)	(-50)	(-127)	-5.6	-2	(-78)	(-50)	(-191)
y	-2	-0.5				0.5	-0.5			

Figure 19: Value of the objective function G

```
G = 7.6307e-06
```

Figure 20: Final frame of the path generation render



Function Generation

For function generation, the x and y components are calculated differently compared to the motion and path generation as shown in the figure below. However, the objective function remains the same. The result of the function is also shown below.

Figure 21: Vector representations for the four precision points

$$f_{jx} = -r_{2x}\cos(\beta_j) + r_{2y}\sin(\beta_j) - r_{3x}\cos(\varphi_j) + r_{3y}\sin(\varphi_j) + r_{4x}\cos(\gamma_j) - r_{4y}\sin(\gamma_j) + r_{2x} + r_{3x} - r_{4x} = 0$$

$$f_{jy} = -r_{2x}\sin(\beta_j) - r_{2y}\cos(\beta_j) - r_{3x}\sin(\varphi_j) - r_{3y}\cos(\varphi_j) + r_{4x}\sin(\gamma_j) + r_{4y}\cos(\gamma_j) + r_{2y} + r_{3y} - r_{4y} = 0$$

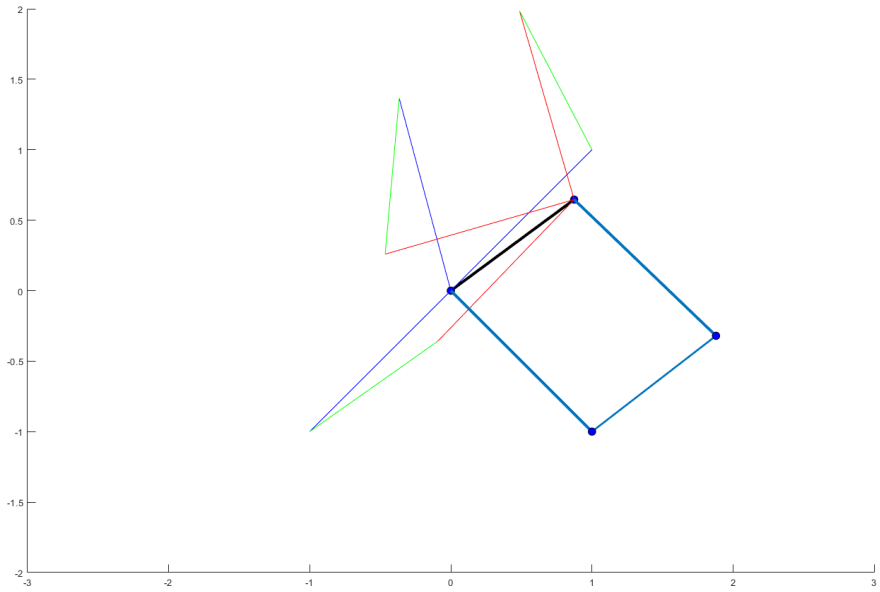
Figure 22: Value of the objective function F

```
F = 5.1408e-06
```

Figure 23: Initial conditions for function generation

Initial Conditions					
	R_3	**R_4**	**φ_1**	**φ_2**	**φ_3**
x (θ)	0.5	1	(-10)	(-70)	(-167)
y	-0.5	-0.5			

Figure 24: Final frame of the function generation render



Conclusion

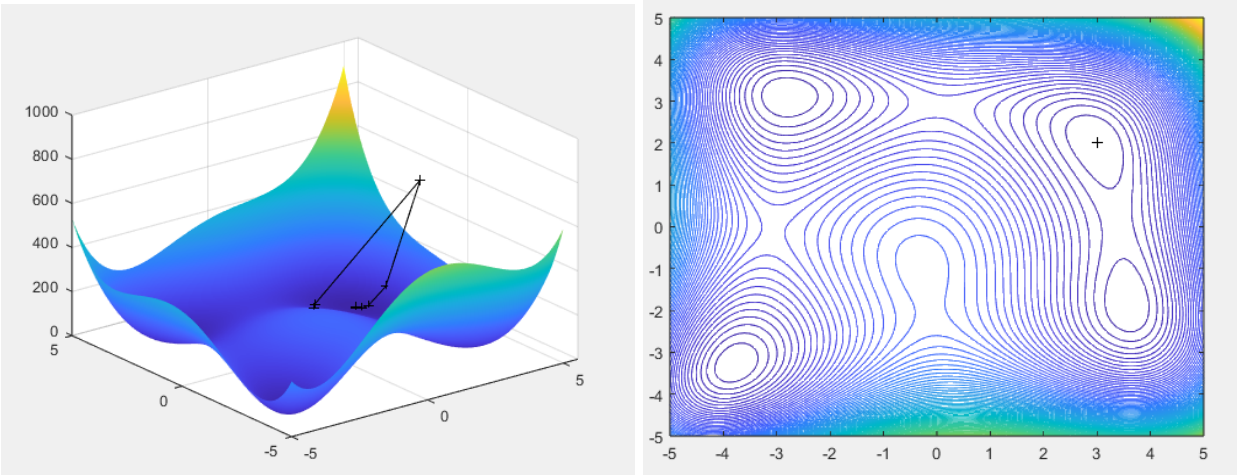
Two optimization problems were solved on Matlab. The first optimization problem was solved using three different methods, steepest descent method, Newton's method, and the function method. The second problem was solved using motion, path, and function generation. The x and y components of the four precision points are determined and was the input of the objective function which represented the length of the linkage bars of a mechanism.

Appendix

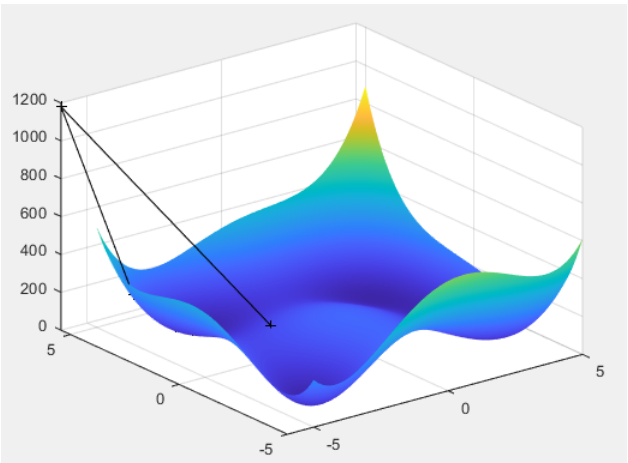
Figure 1: Formulas for the step size, gradient, and the Hessian

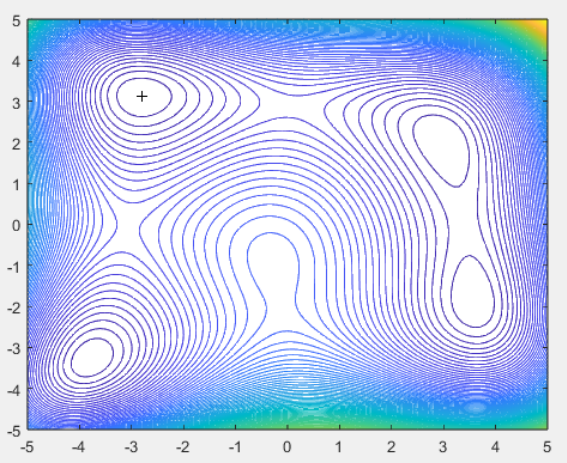
$$h = \left|\frac{\nabla(\mathbf{x})^T\nabla(\mathbf{x})}{\nabla(\mathbf{x})^T\mathbf{H}(\mathbf{x})\nabla(\mathbf{x})}\right|; \quad \text{with } \nabla(\mathbf{x}) = \begin{bmatrix}\frac{\partial f}{\partial x} \\ \\ \frac{\partial f}{\partial y}\end{bmatrix} \quad \text{and } \mathbf{H}(\mathbf{x}) = \begin{bmatrix}\frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2}\end{bmatrix}$$

Figure 2: Path followed and contours for Points 1 - 4
Steepest descent method
Point (0,0)

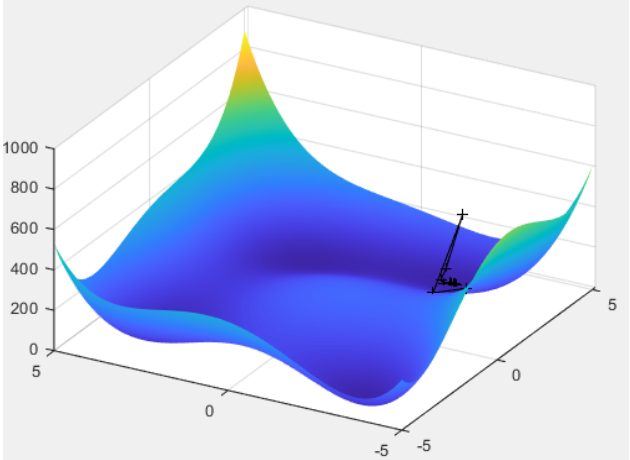


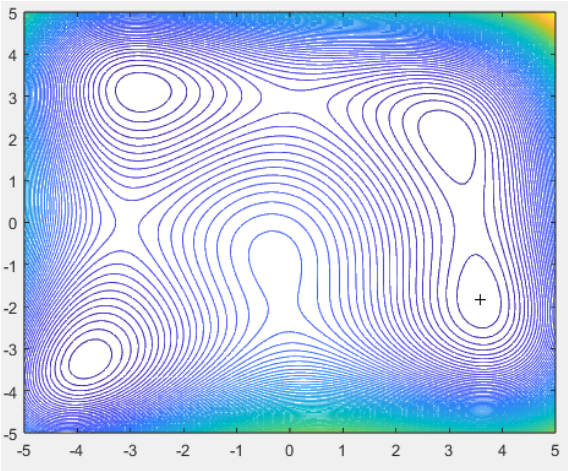
Point (-1,0)



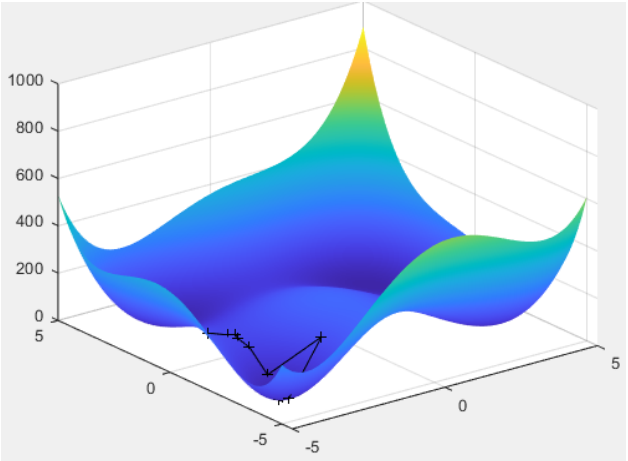


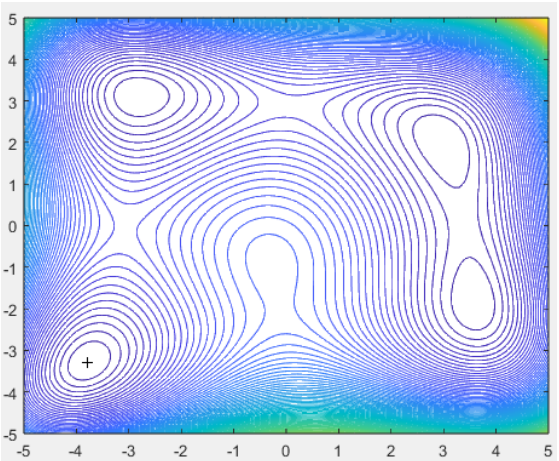
Point (1,-5)





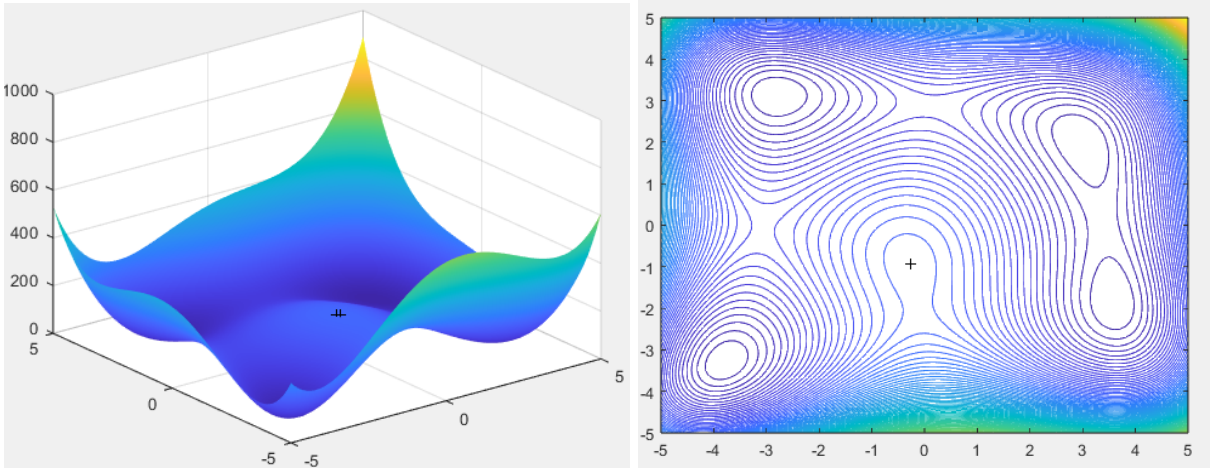
Point(-5,0)



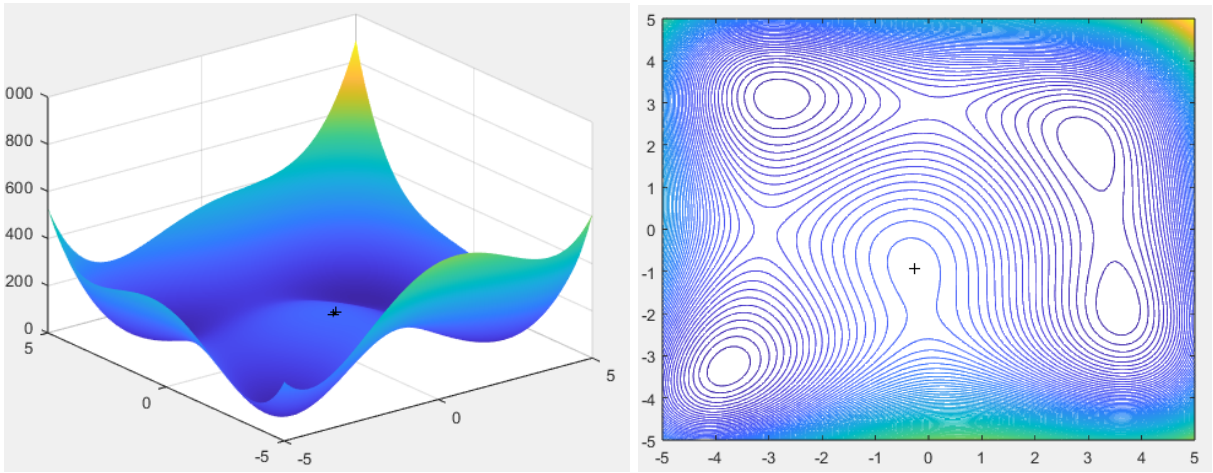


Newton Method

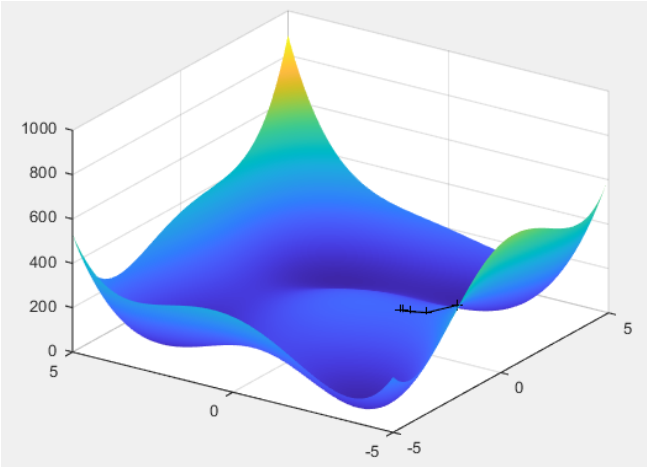
Point (0,0)

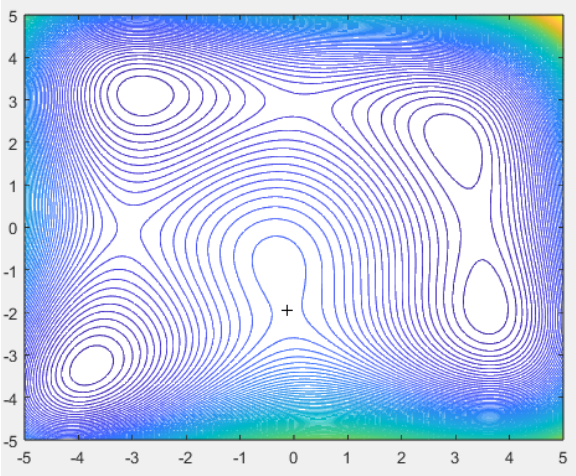


Point (-1,0)



Point (1,-5)





Point (-5,0)