STOCHASTIC AND DETERMINISTIC OPTIMIZATION

By Wilfried ZAKIE
ADEO1

During our deterministic and stochastic optimization classes, we have studied 4 different optimization algorithm. These algorithms were applied on the RosenBrock and Himmelblau functions. The objective is to find the minima of these functions using as parameters the direction d_k and, for some algorithms the step t_k.

The direction d_k is a vector defined by the gradient of the function. As to the step t_k it's a constant obtained mostly using the Hessian matrix if the function.

The algorithms studied are listed below:

- Steepest descent algorithm
- Newton algorithm
- Quasi Newton algorithm
- Quasi-Newton algorithm using Wolf algorithm

We are going to compare the performance of each of them to optimize the Rosenbrock and Himmelblau functions.

I. Summary of the algorithm

1. Steepest descent

The Steepest descent is an algorithm that compute to at each iteration, the next point using the step t_k and direction d_k where:

$$t = \frac{<d_k, d_k>}{<H(x_k).d_k, d_k>}$$

And $d = \nabla f(x_k)$

H is the Hessien matrix of this function.

2. Newton algorithm

Similar to the Steepest descent Algorithm, the newton algorithm takes into account only the direction at each iteration to find the minima of the function.

$$d = -H^{-1}.\nabla f$$

The goal is to find at each iteration k, the linear approximation of the point x_{k+1}.

$$\boldsymbol{x_{k+1} = x_k + d}$$

3. Quasi-Newton Algorithm

In the Quasi Newton algorithm, the step is calculated using θ whose value is:

$$\theta_{k+1} = \theta + \frac{<\Delta,\Delta>}{<\Delta,\sigma>} - \frac{<\theta.\sigma\ ,\ \sigma>}{<\sigma,\ \theta.\sigma>}.\theta$$

Where $\Delta = x_{k+1} - x_k$, $\quad \sigma = \nabla f(x_{k+1}) - \nabla f(x_k)$

The point at the next iteration is:

$$x_{k+1} = x_k - \nabla f(x_k).\theta_k$$

4. Quasi-Newton Algorithm using Wolf

This algorithm use the same principle with the Quasi-Newton Agorithm. But here, the step t_k is used to find the minima. The step is calculated using the wolf algorithm. At each iteration, the Wolf algorithm look for an optimal step that lies in the region of the minima. If the current step is outside this region, some operations are performed in order to make it optimal.

The point at the next iteration is:

$$x_{k+1} = x_k - t_k.\nabla f(x_k).\theta_k$$

In our example, we will take $t_0 = 1$ and $\varepsilon = 0.001$

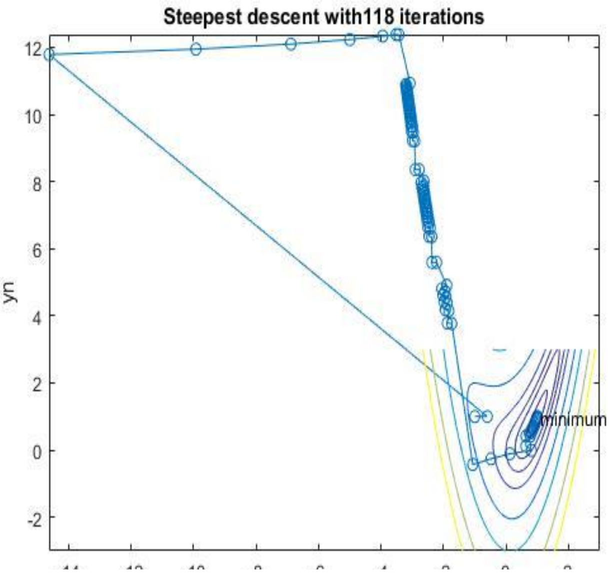
II. Study of Rosenbrock function

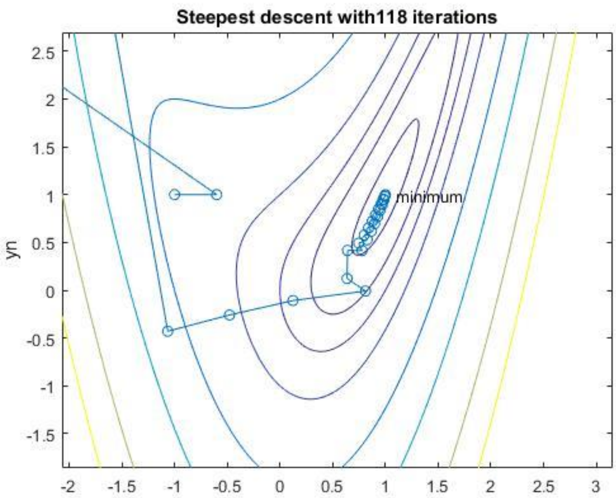
The Rosenbrock function is defined over R² by :

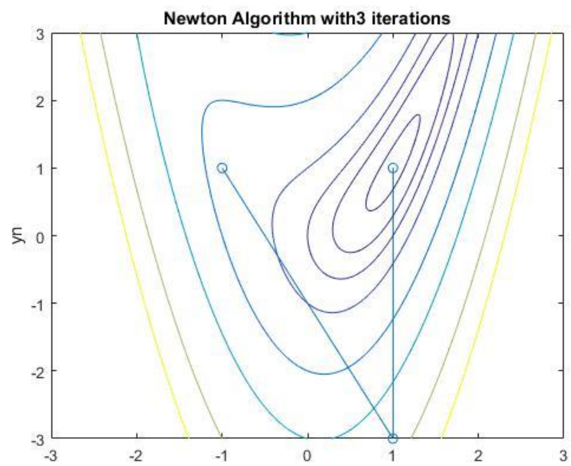
$$f(x, y) = (x^2 - y)^2 + (x - 1)^2$$

Starting from the points (-1,1) and (2,3), we are going to compare the performance of our algorithms and get the number of iterations required to reach the minimal point.

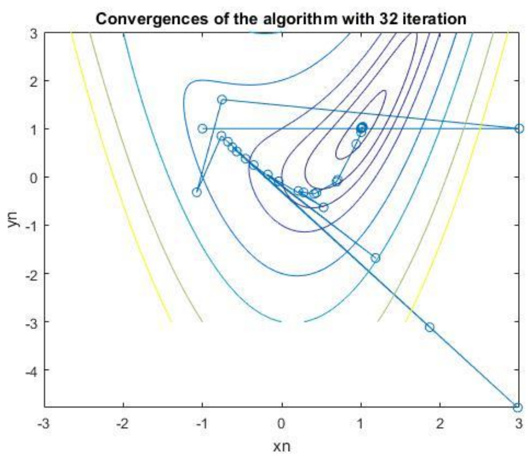
- **For the point (-1,1)**



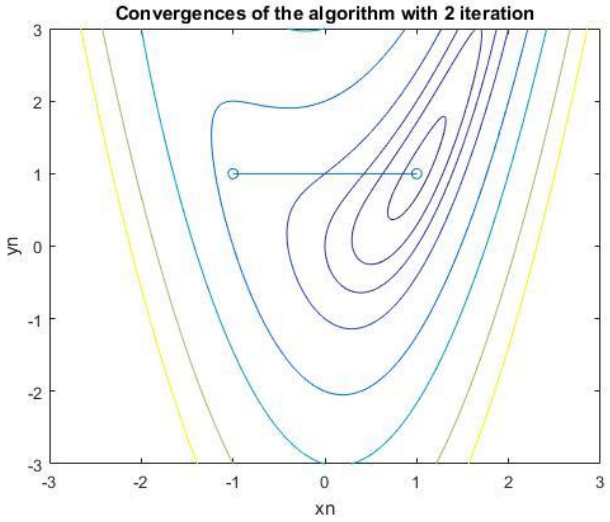




Quasi Newton Algorithm

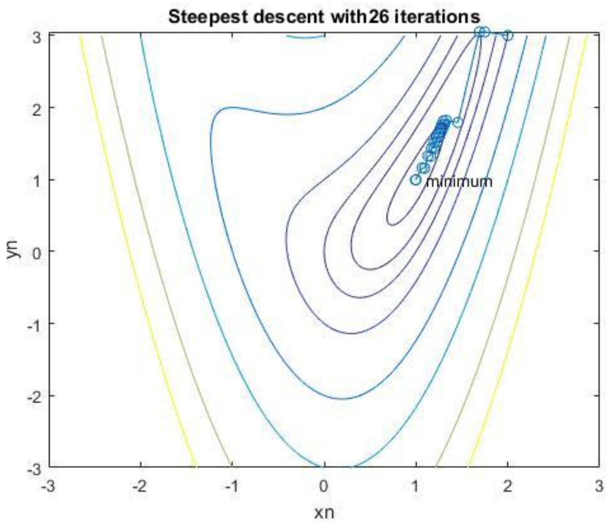


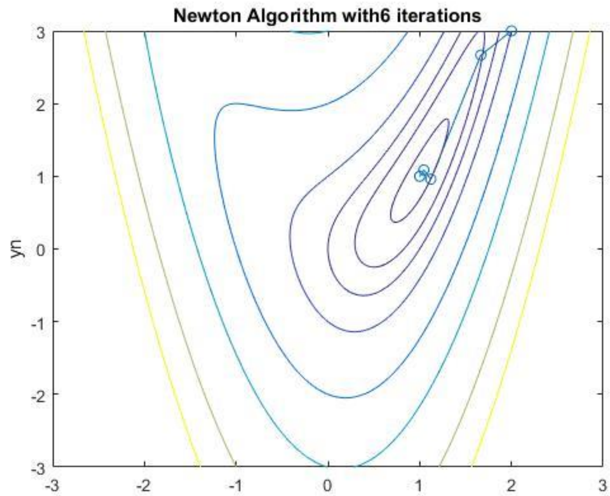
Quasi Newton Algorithm with Wolf



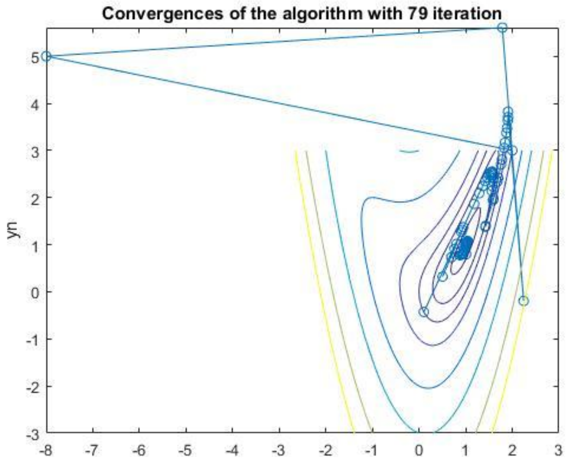
Algorithm	Iterations
Steepest descent Algorithm	118
Newton Algorithm	3
Quasi Newton Algorithm	32
Quasi Newton with Wolf Algorithm	2

- **<u>For the point (2,3)</u>**

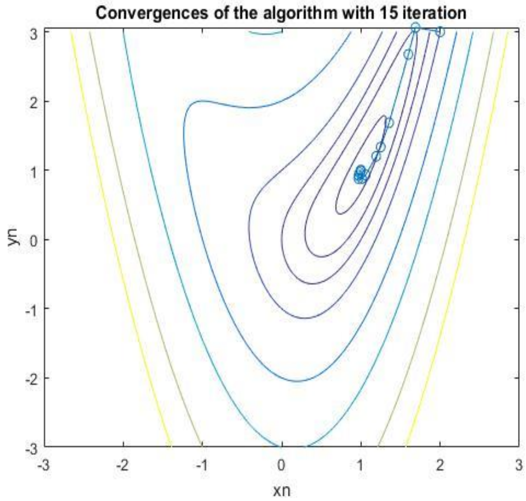




Quasi Newton Algorithm



Quasi Newton Algorithm with Wolf



Algorithm	Iterations
Steepest descent Algorithm	26
Newton Algorithm	6
Quasi Newton Algorithm	75
Quasi Newton with Wolf Algorithm	15

III. Study of the Himmelblau function

The Himmelblau function is a function defined over R² by:

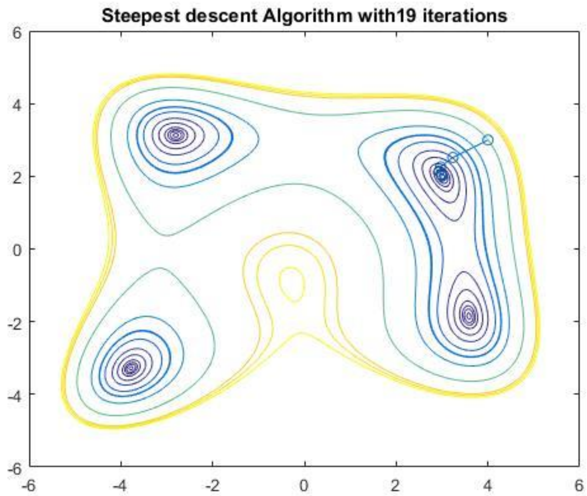
$$f(x, y) = (x^2 + y - 11)^2 + (x + y^2 - 7)^2$$

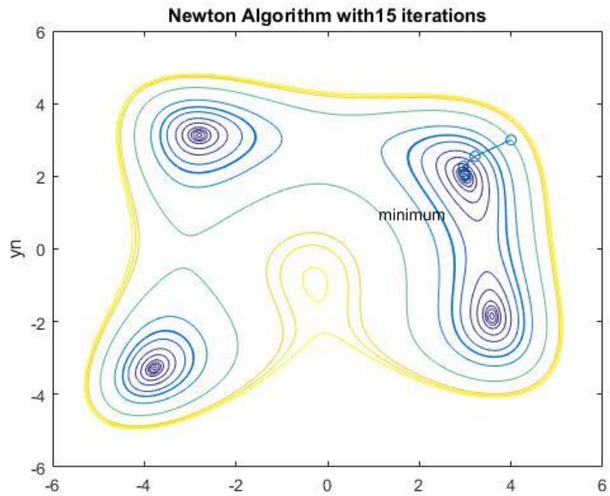
The same algorithms will be Applied on this function and the result will be compared.

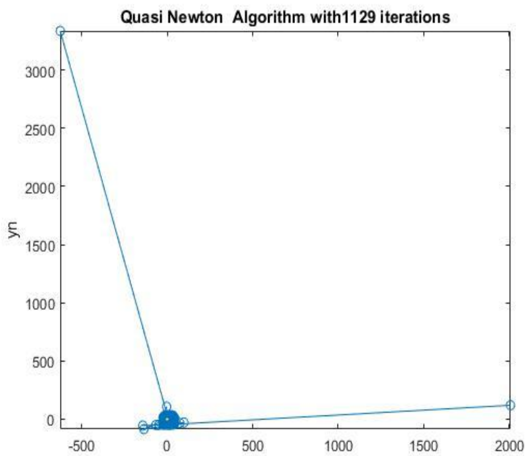
The points used are the following:

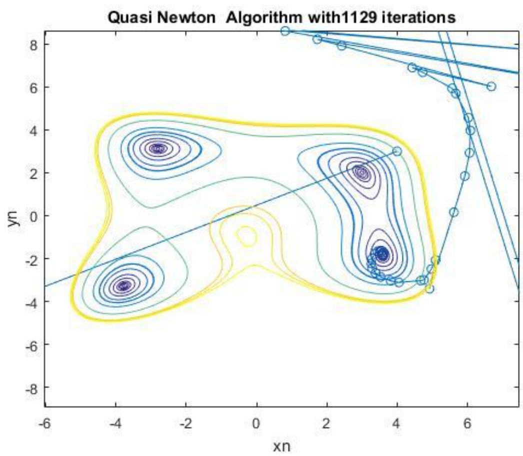
- (3,3)
- (-4,1)
- (-2,-3)
- (7,-6)

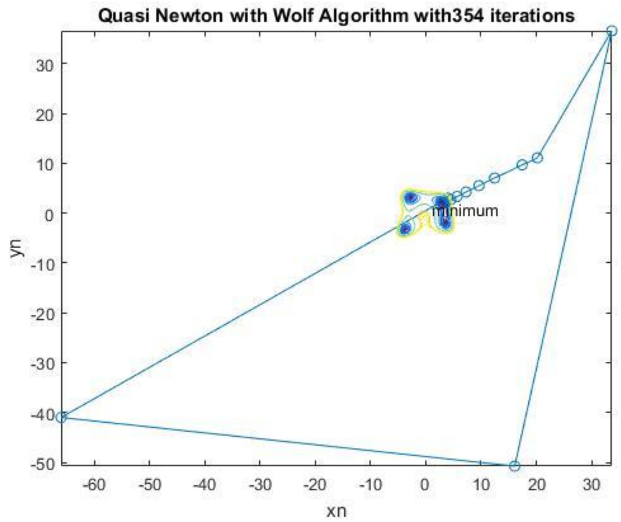
- **For the point (4,3)**

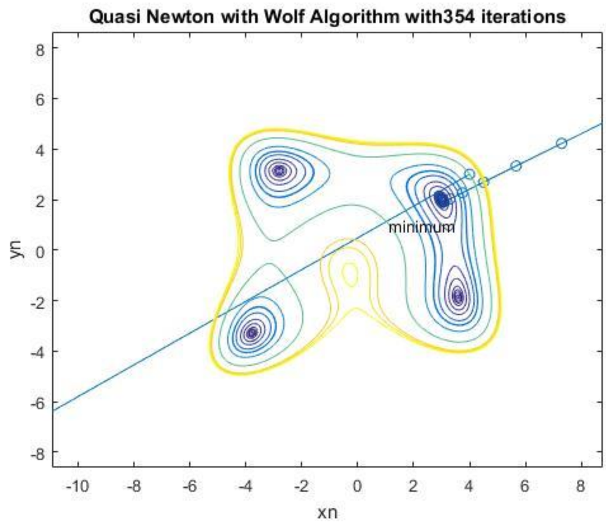






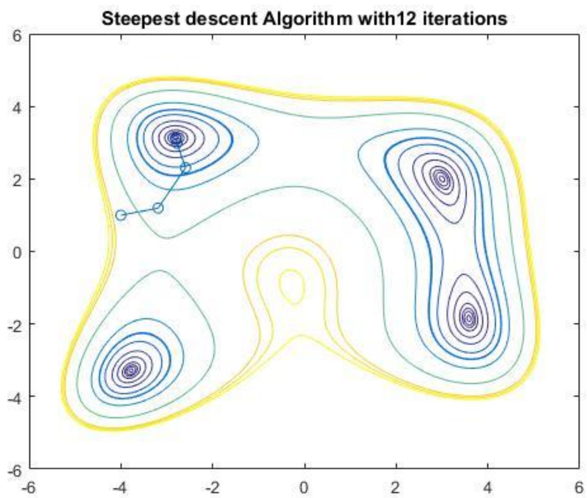


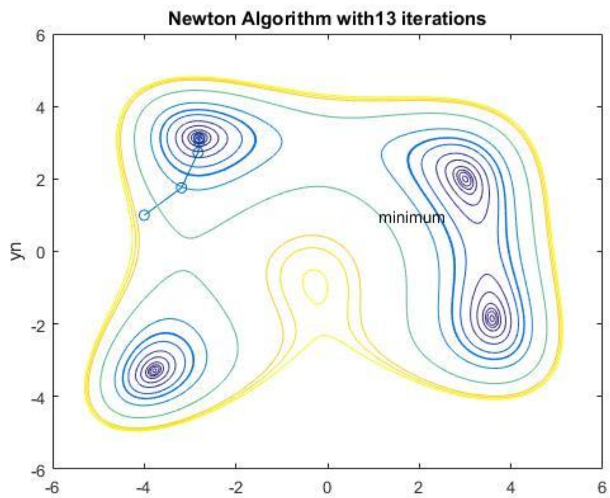


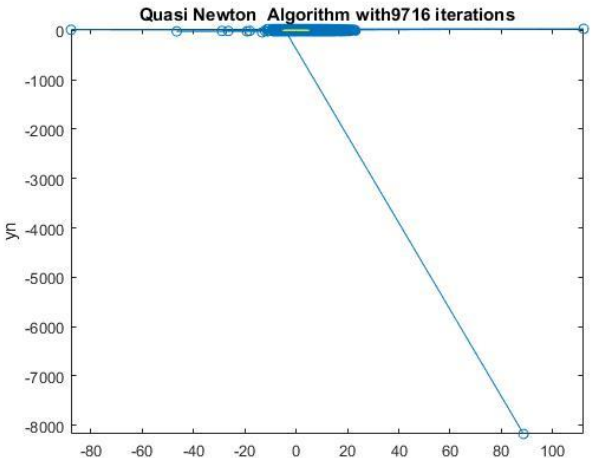


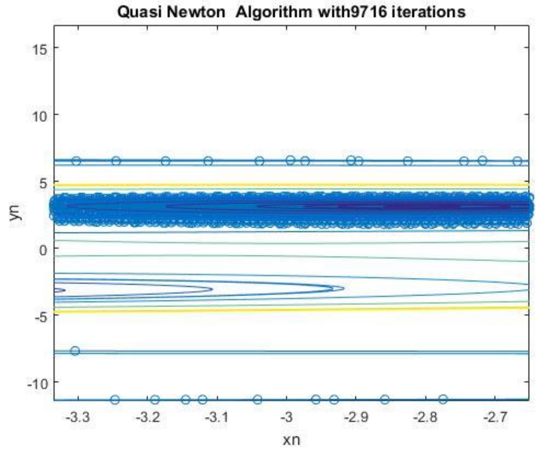
Algorithm	Iterations
Steepest descent Algorithm	19
Newton Algorithm	15
Quasi Newton Algorithm	1129
Quasi Newton with Wolf Algorithm	354

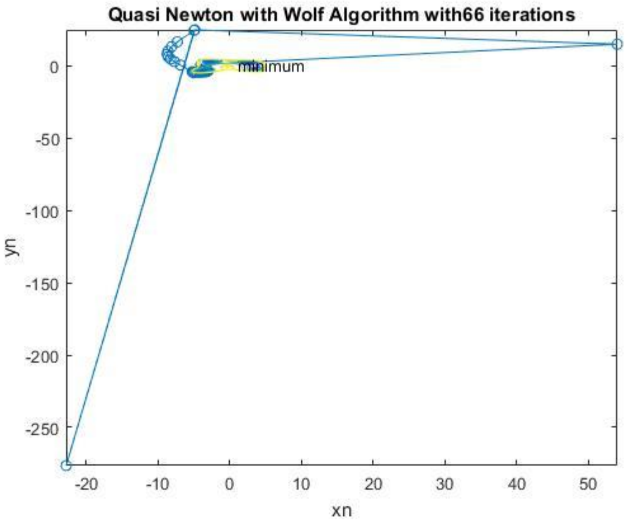
- **For the point (4,3)**

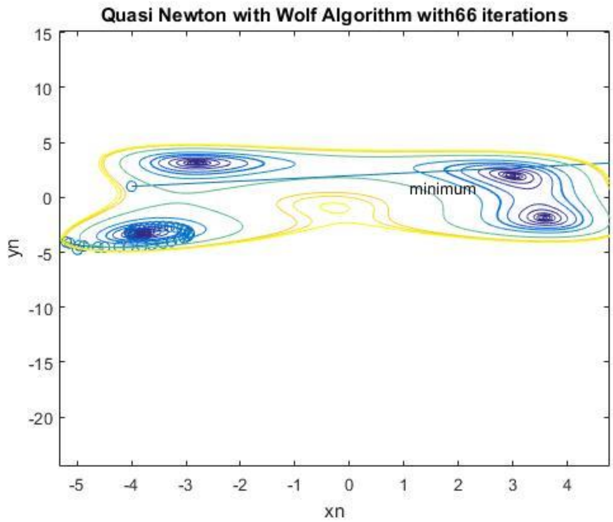






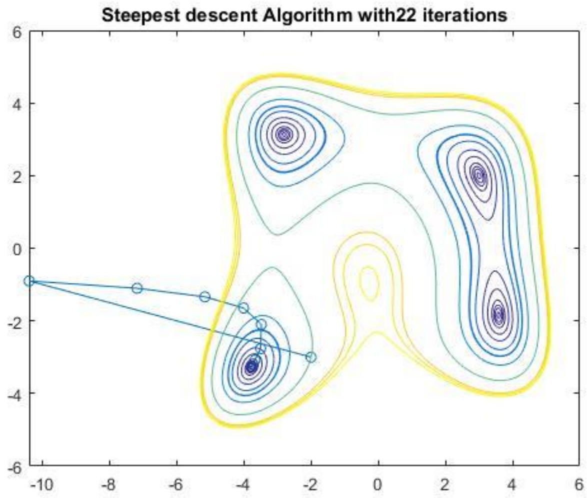


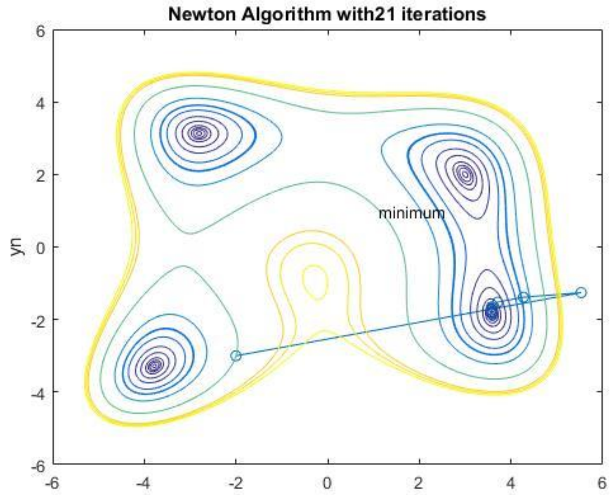


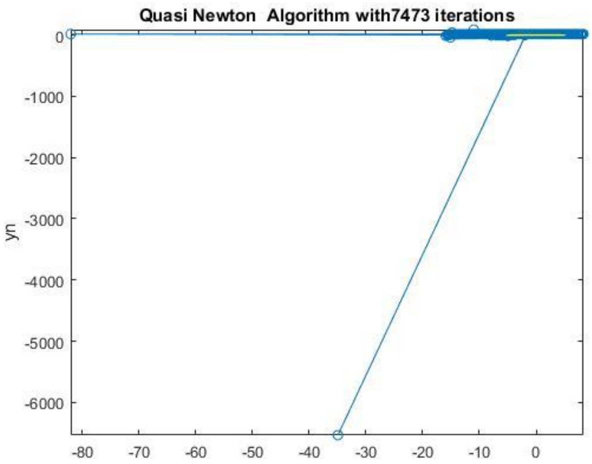


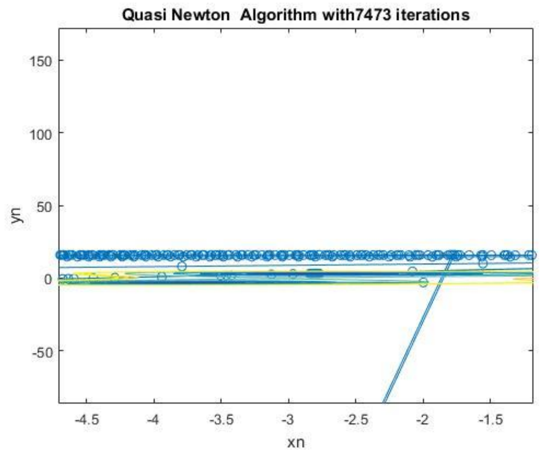
Algorithm	Iterations
Steepest descent Algorithm	12
Newton Algorithm	13
Quasi Newton Algorithm	9716
Quasi Newton with Wolf Algorithm	66

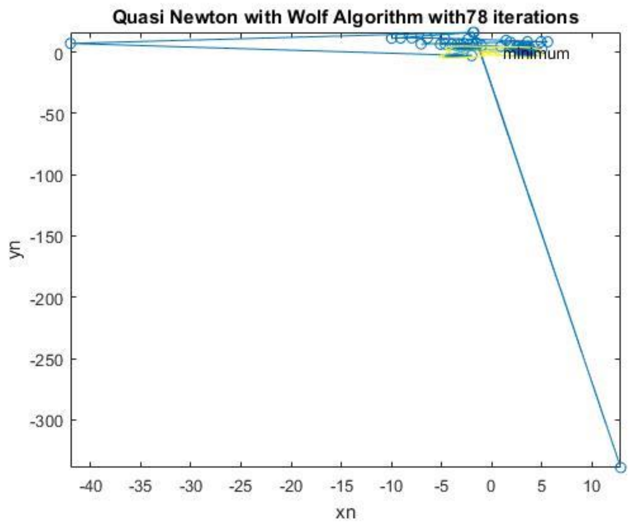
- **For the point (-2-3)**

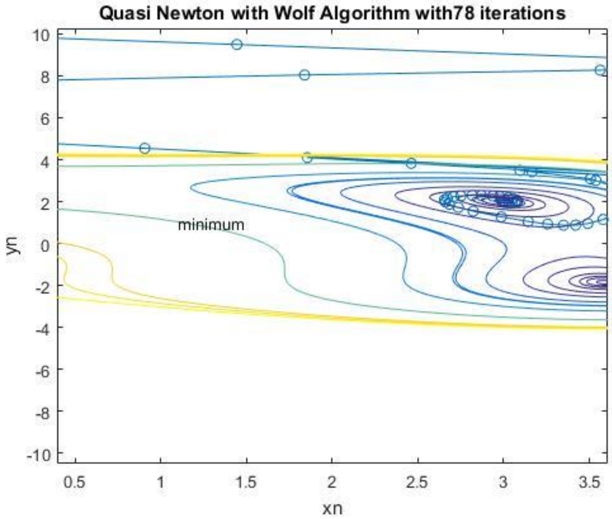






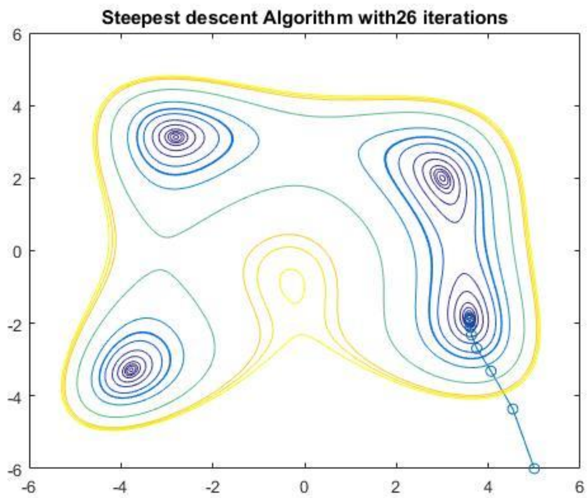


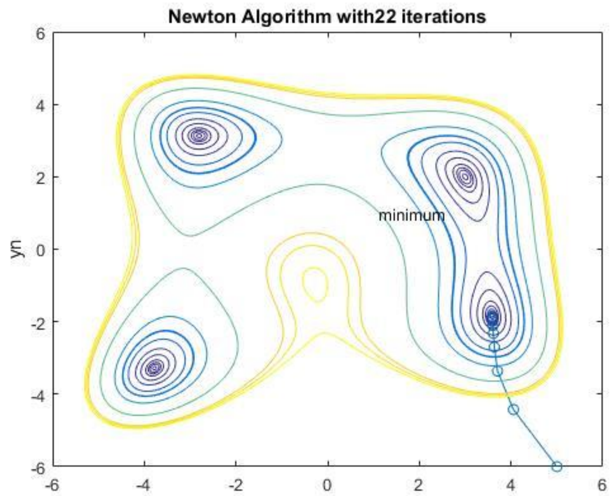


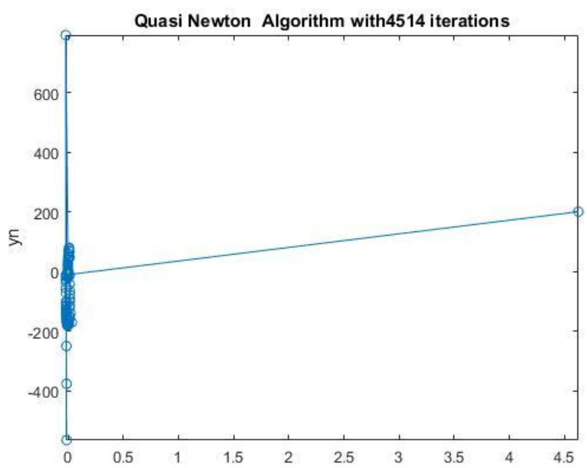


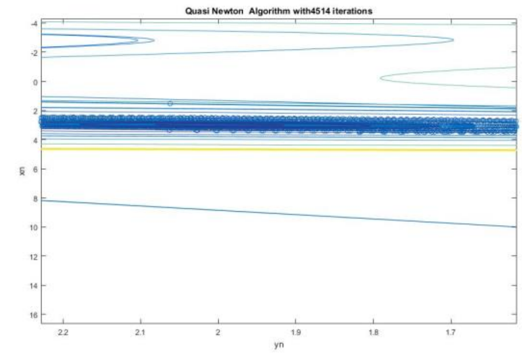
Algorithm	Iterations
Steepest descent Algorithm	22
Newton Algorithm	21
Quasi Newton Algorithm	7473
Quasi Newton with Wolf Algorithm	78

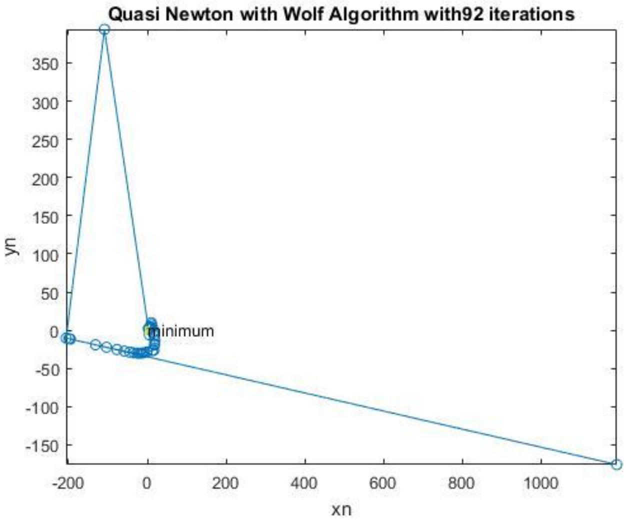
- **<u>For the point (-2-3)</u>**

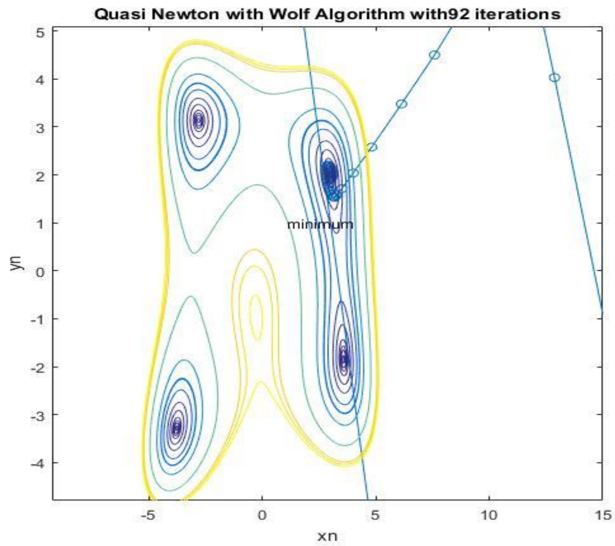












Algorithm	Iterations
Steepest descent Algorithm	22
Newton Algorithm	26
Quasi Newton Algorithm	4514
Quasi Newton with Wolf Algorithm	92

CONCLUSION

We notice that the performance of the four algorithms firstly depends on the function on which they are applied and secondly on the starting point. For example, the steepest descent algorithm is more efficient to optimize the Himmelblau function than the Rosenbrock function. Also the Quasi Newton algorithm requires a lot of iterations before getting to the minimal point. We get more efficient results by combining these algorithm with Wolf function.

Looking at the good aspect as well as the drawbacks of our algorithms, one should applied them depending on the problem and compare them in order to make the best optimization possible.