Proceedings of International Conference on Scientific Research in Natural and Social Sciences
Hosted online from Toronto, Canada.
**Date:** 5th April, 2023
ISSN: 2835-5326 **Website:** econferenceseries.com

# SOLUTION OF MULTI-DIMENSIONAL OPTIMIZATION PROBLEMS USING NEWTON'S METHOD

Feruza Rakhimova Saidovna,
senior teacher of Tashkent University of Information Technologies, Department of Mathematics

Tohirov Quvonchbek
student of the Faculty of Software Engineering of the University of Tashkent Information Technologies

One-variable function optimization is one of the simplest optimization problems. However, it occupies an important place in the theory of optimization. The reason is that one-parameter optimization problems are often encountered in engineering practice, and in addition, complex multidimensional widely used in optimization problems.

The function F(x) is very complicated in practice, and it may be impossible to find its extremum analytically by taking its derivative, or it may require a lot of time. In practical problems, in many cases, it is enough to find the exact solution with some error. Therefore, it is an urgent problem to find a solution with sufficient error in numerical methods.Let's say that it is required to find the minimum of the function f(x) in the interval [a;b]. Optimization methods using only the values of the function f(x) are called 0-order methods. These include:

1) General search method;
2) method of unimodal functions;
3) The method of dividing the interval into two equal parts;
4) Golden forty method;

All of these methods are based on reducing the interval in which the function is optimally searched, and are called interval exclusion methods.

If the function f(x) is smooth enough, for example, it has continuous first and second order derivatives, then the speed of convergence of numerical methods can be increased.

ISSN: 2835-5326 Website: econferenceseries.com

Optimization methods that use 1-2- ,…….. order derivatives of the function are called 1-2-,….. order methods, respectively.

Suppose the function f(x) is twice differentiable. We know that the minimum condition of such a function is:

$$f'(x^*) = 0$$

A condition is a sufficient condition.

So, $f'(x^*) = 0$we solve the equation numerically. We give the initial approximation of $X_k$ (we choose it as close to the solution as possible) and at this point we expand it to the Taylor series.

$$f(x) \approx \bar{f}(x) = f(x_k) + f'(x_k)(x - x_k) + \frac{1}{2!}f''(x_k)(x - x_k)^2$$

If it is $f''(x_k) \neq 0$, then f(x) has a unique stationary point. To find this point, we set $\hat{f}'(x)$ the derivative equal to zero:

$$f'(x^*) = 0; \hat{f}'(x)2\left(f(x) + f'(x)(x - x_k) + \frac{1}{2!}\left((x_k)(x - x_k)^2\right)'\right) = 0$$

$$=> 0 + f^1(x_k) + \frac{1}{\alpha} * 2f''(x_k)(x - x_k) = 0 => x = x_k - \frac{f'(x_k)}{f''(x_k)}$$

we accept the found solution as the k+1-th approach to the minimum of x, as a result we get the following iterative formula:

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$$

Note: The formula given above is used to solve the equation f(x)=0

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$$

Not to be confused with Newton's formula.

This algorithm has 2 disadvantages:

1) $f'(x) = 0$the equation can determine not only the minimum, but also the maximum.

2)$\hat{f}(x)$ the model function may be significantly different from the function f(x) being optimized.

3)$X_{k+1}$-$x_k$- the step can be very large.

That is why we check the condition $f(x_{k+1})<f(x_k)$ at each step to check that we are going to the minimum.If this condition is met, then we will proceed to the next step. If $f(x_{k+1})<f(x_k)$ and $f'(x)(x-x_k)<0$ , then f(x) must decrease from$x_k$ to $x_{k+1}$ at the beginning of the function. can be found, for example.

$$x'_{k+1}=\frac{x_{k+1}+x_k}{2}$$

(*) as can be seen from the formula, $f'(x_k)(x-x_k)$the expression is negative only and only $f''(x_k)$ >0 .This means that an optimal step direction is guaranteed to obtain a Newton step.

On the other hand, if $f''(x)<0$ and $f'(x_k)(x-x_k)$ >0, then f(x) increases at the beginning when passing from $x_k$ to $x_{k+1}$, so the step should be taken in the opposite direction.

As a criterion for stopping iteration (approximator) in optimization.

$$\left|\frac{f'(x_{k+1})}{f(x_k)}\right|<\varepsilon$$

The condition can be accepted, $\varepsilon$ - certainty given in advance. This method is called Buton or Newton-Raphson method. In some problems, it is difficult to get the derivatives of the function f(X), in such cases, Newton's method can be modified. For this, we choose the initial approximation $X_k$ and the step h. Let's look at the points $X_k$-h, ,$x_k$, $x_k$+h. In that case, the derivatives f'(X) and f''(x) can be replaced by the following approximate formulas.

$$f'(x_k)\approx\frac{f(x_k+h)-f(x_k-h)}{2h} \quad \text{or} \quad f'(x)=\lim_{h\to 0}\frac{f(x+h)-f(x-h)}{h\cdot 2}$$

=>Leaving the lim difference, we get the following approximate formulas:

$$f'(x_k)\approx\frac{f(x_k+h)-f(x_k-h)}{2h};$$

$$f''(x_k)\approx\frac{f'(x_k+h)-f'(x_k-h)}{2h}=\frac{\frac{f(x_k+h)-f(x_k)}{h}-\frac{f(x_k)-f(x_k-h)}{h}}{2h}=>$$

$$f''(x_k)\approx\frac{f(x_k+h)-2f(X_k)+f(X_k-h)}{2h^2}$$

In that case, if we put (*) in the above formula, we will get the iterative formula as a result:

$x_{k+1} = x_k - \frac{f(x_k+h)-f(x_k-h)}{f(x_k+h)-2f(X_k)+f(X_k-h)}$ *h

This formula is called Cauzi Newton or Newton's modified method.

For example: it is required to find f(x) $=x^3-x^2 \rightarrow min\ \varepsilon = 0{,}001$with error.

We find the first and second derivatives of the given function:

$$f'(x) = 3x^2 - 2^x \quad f''(x) = 6x - 2$$

According to the graph of the function, the local minimum lies between 0.5 and 1. (In fact, it is possible to quickly find the minimum of such a function analytically. It has a local minimum at exactly 2/3. to check.) We will complete the necessary calculations in the Excel program package and present the necessary information in the following table.

The table will be written.\

In the 4th step of the iteration, we determine the minimum point $x_{min} = 0.66667$ whose error is less than 0.00000021/

Disadvantages of Newton's method:

1) The method requires a sufficiently good initial approximation.
2) First and second order derivatives need to be given analytically.
3) In Newton's method, there is no barrier that prevents the iterative method from drifting towards maximum or turning points.

Let's say that the n-dimensional function f(x) =f($x_1,x_2,\ldots\ldots\ldots x_n$) is given in $R^n$ space. Let the unconditional optimization problem be set:

F($x_1,x_2\ldots\ldots\ldots\ldots x_n$) -> min

The necessary condition for the extremum for a multivariable function is as follows:

$$\begin{cases} \frac{\partial f(x_1x_1, x_2 \ldots\ldots x_n)}{\partial x_1} = 0 \\ \frac{\partial f(x_1x_1, x_2 \ldots\ldots x_n)}{\partial 2} = 0 \\ \frac{\partial f(x_1x_1, x_2 \ldots\ldots x_n)}{\partial x_3} = 0 \end{cases}$$

Takes the form. We enter the following designations:

$$\nabla f(x_1x_1, x_2 \ldots\ldots x_n) = grad\, f(x) = \left(\frac{\partial f(x_1x_1,x_2 \ldots\ldots x_n)}{\partial x_1}, \ldots \ldots, \frac{\partial f(x_1x_1,x_2 \ldots\ldots x_n)}{\partial x_n}\right)^T$$

X=$(x_1 x_1, x_2 \ldots\ldots x_n)$ ; $x_k$ =( $x_1^{(k)}$, $x_2^{(k)}$............$x_n^{(k)}$); $x^T$= $\begin{pmatrix} x_1 \\ x_2 \\ \ldots\ldots \\ x_n \end{pmatrix}$ - x is the transpose of a vector.

As the norm of any matrix A, one can take the largest of the modules of the eigenvalues.

$$\|A\| = \max_{1\le k\le n} |\lambda_k|$$

The eigenvalues of the matrix are the solutions of $det(A - \lambda E) = 0$ equations. There are several ways to choose the matrix norm.For example, it is possible to enter norm $\|A\|_1 = \max_{1\le i\le m} \sum_{j=1}^{n} |a_{ij}|$ on rows or norm $\|A\|_2 = \max_{1\le i\le m} \sum_{j=1}^{n} |a_{ij}|$ on columns and other norms.

$H(x_k)$ - the value of the Hessian matrix consisting of the second-order eigenderivatives of the given function at point $x_k$

Hessian matrix for a 2-variable function:H($x_k$) = $\begin{pmatrix} f''_{xx} & f''_{xy} \\ f''_{yx} & f''_{yy} \end{pmatrix}$

Hessian matrix for a 3-variable function:H($x_k$) = $\begin{pmatrix} f''_{xx} & f''_{xy} & f''_{xz} \\ f''_{yx} & f''_{yy} & f''_{yz} \\ f''_{zx} & f''_{zy} & f''_{zz} \end{pmatrix}$;

Etc.

such as one-variable optimization problems $\nabla f = 0$ we solve the equation numerically. We give the initial $x_k$ approximation (we choose it as close to the solution as possible) and at this point we expand the function into a Taylor series.

$$\varphi(x) = f(x_k) + \nabla f(x_k)(x - x_k) + \frac{1}{2!}(x - x_k)^T H(x_k)(x - x_k) \quad (**)$$

If the function f(x) has a minimum, then the function reaches a minimum (**) at the point where the quadratic form reaches a minimum.

If the objective function If the Hessian matrix $H(x_k)$ is positive defined from the point $x_k$ , then the $x_k$ point that reaches the minimum of the function $\varphi(x)$ is the only one, and it is found from the condition that the gradient is equal to the vector 0

$\nabla\varphi = \nabla f(x_k)+H(x_k)(x - x_k)=0$

In this formula, we accept the solution found for x as the k+l-approach to the minimum of x, as a result, we get the following iterative formula:

$$x_{k+1} = x_k - H^{-1}(x_k)\nabla f(x_k)$$

In this case, the $H^{-1}(x_k)$-matrix is the inverse matrix of the Hesse matrix. The search direction optimization algorithm determined from this relationship is called Newton's method. The direction

$P_k = -H^{-1}(x_k)\nabla f(x_k)$ is called Newton's direction. This direction forms a non-crossing angle with the gradient vector. In Newton's method, the minimum approximation to a point depends on the choice of initial approximation (point). If the objective function is strongly convex and $\tilde{x} \epsilon D_l(f)$ is arbitrary, the Hessian matrix H(x) of the objective function for the points

$$\|H(x) - H(\tilde{x})\| \leq L \cdot |x - \tilde{x}| \ , L > 0$$

If the condition is fulfilled and the initial approximation is chosen close enough to the minimum point, then the algorithm of Newton's method will have a quadratic approximation speed, that is, the following estimate will be appropriate:

$|x_k - x^*| < C|x_{k-1} - x^*|^2, C > 0$

If the objective function is not strongly convex or the initial approximation is far from the sought point, Newton's method may deviate.

In the problems of finding the minimum of arbitrary quadratic functions with a positive definite Hessian matrix, Newton's method gives a solution in one iteration, regardless of how the initial approximation point is chosen.

The speed of quadratic approximation, as well as the ability to control the sufficient condition of reaching the minimum of the objective function using the Hessian matrix in each k-iteration, show that this algorithm has high efficiency. However, several problems arise in its practical application.

1) In each k-iteration, the $H(X_k)$ Hesse matrix should be kept positive definite, otherwise it is possible that the vector Pk will not be directed towards the minimum.

2) $H(X_k)$ - matrix is an eigenmatrix and its inverse matrix may not exist

3) One of the most important problems is to calculate the n x n matrix in each iteration and find its inverse, which requires a lot of calculations and time when the number of variables n is large.

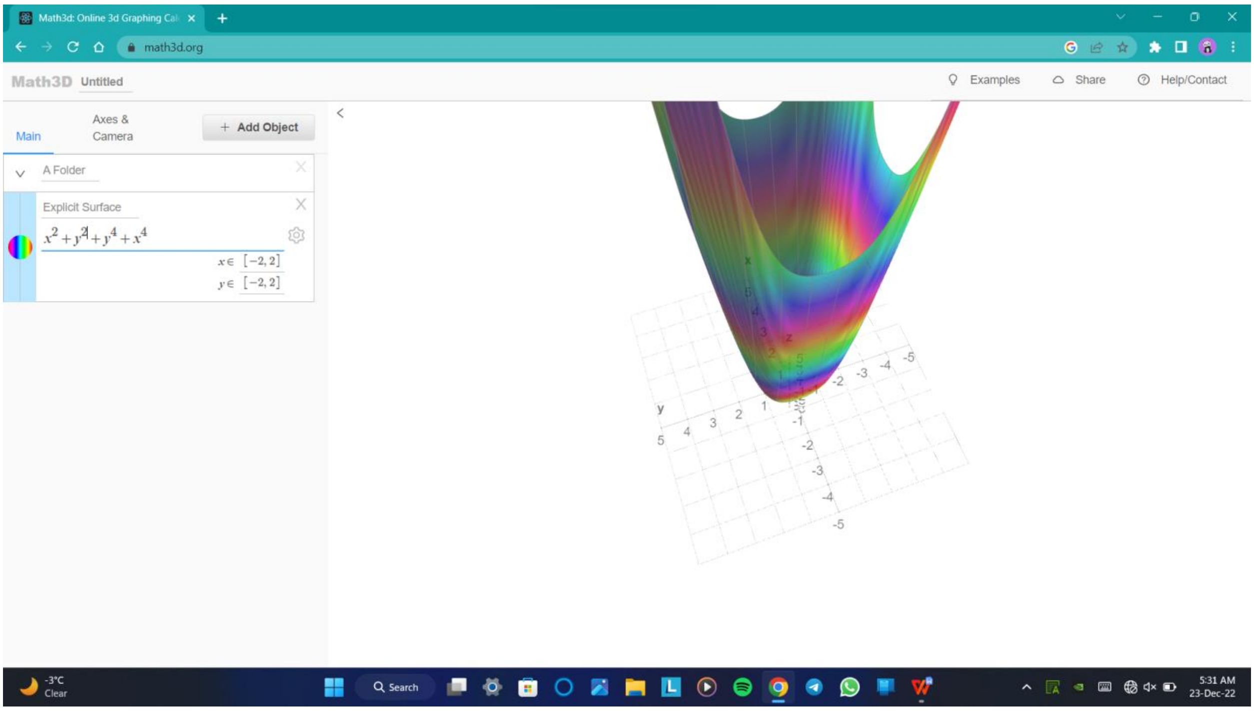
Therefore, there are various modifications of Newton's method to avoid or solve such problems.

Let's look at the following examples:

**Example 1.**

$z = x^4 + y^4 + x^2 + y^2$ Let the two-variable function be given.

Using the graph of the function, we can choose the point $M(-\frac{1}{2};\frac{1}{2})$ close to the minimum value. We use the above formulas in sequence. We find the first and second derivatives of the function with respect to x and y:




$$\begin{cases} \frac{\partial z}{\partial x} = 4x^3 + 2x \\ \frac{\partial z}{\partial y} = 4y^3 + 2y \end{cases} \Rightarrow \frac{\partial^2 z}{\partial x^2} = 12x^2 + 2 \ , \ \frac{\partial^2 z}{\partial x \partial y} = 0 \ , \ \frac{\partial^2 z}{\partial y^2} = 12y^2 + 2$$

$$H = \begin{vmatrix} 12x^2 + 2 & 0 \\ 0 & 12y^2 + 2 \end{vmatrix}$$

ISSN: 2835-5326 **Website:** econferenceseries.com

$H(\frac{-1}{2};\frac{1}{2})=\begin{pmatrix}5 & 0\\ 0 & 5\end{pmatrix}=25>0$ positive detected

$$\Delta f(x,y)=\left(\frac{\partial f(x,y)}{\partial x},\frac{\partial f(x,y)}{\partial y}\right)^T=\begin{pmatrix}4x^3+2x\\4y^3+2y\end{pmatrix}=\begin{pmatrix}-\frac{3}{2}\\ \frac{3}{2}\end{pmatrix}$$

$$detH^{-1}\left(\frac{-1}{2};\frac{1}{2}\right)=\frac{1}{25}\begin{pmatrix}5 & 0\\ 0 & 5\end{pmatrix}=$$

$$x_2=\left(\frac{-1}{2};\frac{1}{2}\right)-\frac{1}{25}\begin{pmatrix}5 & 0\\ 0 & 5\end{pmatrix}\begin{pmatrix}-\frac{3}{2}\\ \frac{3}{2}\end{pmatrix}=\left(\frac{-1}{2};\frac{1}{2}\right)-\left(-\frac{3}{10};\frac{3}{10}\right)=\left(-\frac{1}{5};\frac{1}{5}\right)$$

$$H\left(\frac{3}{2};2\right)=\begin{vmatrix}5 & 0\\ 0 & 6\end{vmatrix}$$

We calculate the norm of the Hessian matrix and compare it with the given value of $\varepsilon=0.1$, if the norm of the matrix is greater than $\varepsilon=0.1$, then we find $x_k$ .

$$\left|H\left(\frac{-1}{2};\frac{1}{2}\right)-H\left(-\frac{1}{5};\frac{1}{5}\right)\right|=\left|\begin{pmatrix}5 & 0\\ 0 & 5\end{pmatrix}-\begin{pmatrix}\frac{62}{25} & 0\\ 0 & \frac{62}{25}\end{pmatrix}\right|=\begin{vmatrix}\frac{63}{25} & 0\\ 0 & \frac{63}{25}\end{vmatrix}$$

2,52>0,1

Using the above formulas again, we find the following point:

$$H\left(\frac{-1}{5};\frac{1}{5}\right)=\begin{vmatrix}\frac{62}{25} & 0\\ 0 & \frac{62}{25}\end{vmatrix}=\frac{62^2}{625}.$$

$$\Delta f\left(-\frac{1}{5};\frac{1}{5}\right)=\left(\frac{\partial f(x,y)}{\partial x},\frac{\partial f(x,y)}{\partial y}\right)^T=\begin{pmatrix}4x^3+2x\\4y^3+2y\end{pmatrix}=\begin{pmatrix}-\frac{54}{125}\\ \frac{54}{125}\end{pmatrix}$$

$$det\left(H^{-1}\left(-\frac{1}{5};\frac{1}{5}\right)\right)=\frac{625}{62^2}\begin{pmatrix}\frac{62}{25} & 0\\ 0 & \frac{62}{25}\end{pmatrix}$$

ISSN: 2835-5326 Website: econferenceseries.com

$$x_2=\left(-\frac{1}{5};\frac{1}{5}\right)-\frac{625}{62^2}\begin{pmatrix}\frac{62}{25} & 0\\ 0 & \frac{62}{25}\end{pmatrix}\begin{pmatrix}-\frac{54}{125}\\ \frac{54}{125}\end{pmatrix}=\left(-\frac{1}{5};\frac{1}{5}\right)-\left(-\frac{27}{155};\frac{27}{155}\right)$$
$$=\left(-\frac{4}{155};\frac{4}{155}\right)$$

We calculate the norm of the Hessian matrix and compare it with the given value $\varepsilon=0.1$ , if the norm of the matrix is greater than $\varepsilon=0.1$ , then we find $x_k$ . We continue this process until $\varepsilon=0.1$ is less than the norm of the matrix, and we take the last $x_k$ as the optimal point.

$$\left|H\left(-\frac{1}{5};\frac{1}{5}\right)-H\left(-\frac{4}{155};\frac{4}{155}\right)\right|=\left|\begin{pmatrix}2{,}48 & 0\\ 0 & 2{,}48\end{pmatrix}-\begin{pmatrix}2{,}008 & 0\\ 0 & 2{,}008\end{pmatrix}\right|=\left|\begin{pmatrix}0{,}472 & 0\\ 0 & 0{,}472\end{pmatrix}\right| \qquad 0{,}472>0{,}1$$

$$H\left(-\frac{4}{155};\frac{4}{155}\right)=\begin{pmatrix}2{,}008 & 0\\ 0 & 2{,}008\end{pmatrix}=4{,}03 \qquad \Delta f\left(-\frac{4}{155};\frac{4}{155}\right)=\begin{pmatrix}-0{,}051\\ 0{,}051\end{pmatrix}$$

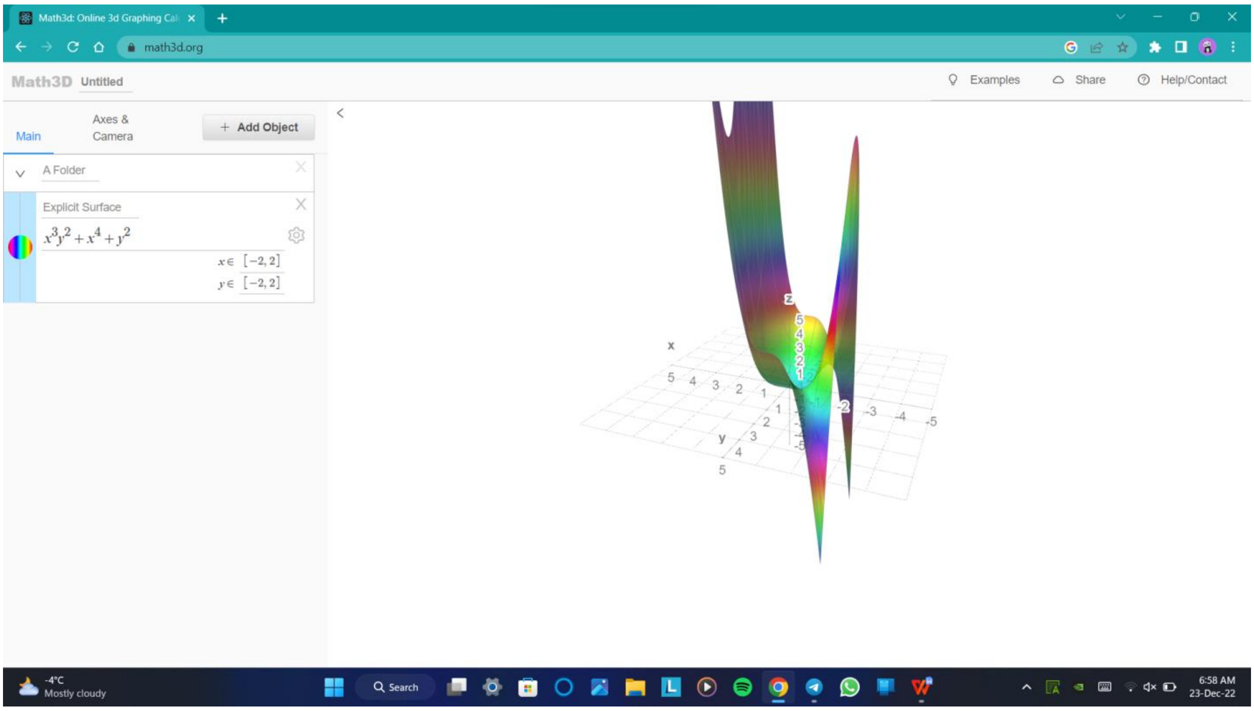
$$det\left(H^{-1}\left(-\frac{4}{155};\frac{4}{155}\right)\right)=\frac{1}{4{,}03}\begin{pmatrix}2{,}008 & 0\\ 0 & 2{,}008\end{pmatrix}$$
$$x_2=(-0{,}026;0{,}026)-\frac{1}{4{,}03}\begin{pmatrix}2{,}008 & 0\\ 0 & 2{,}008\end{pmatrix}\begin{pmatrix}-0{,}051\\ 0{,}051\end{pmatrix}$$
$$=(-0{,}026;0{,}026)-(-0{,}025;0{,}025)=(-0{,}001;0{,}001)$$

$$|H(-0{,}026;0{,}026)-H(-0{,}001;0{,}001)|=\left|\begin{pmatrix}2{,}008 & 0\\ 0 & 2{,}008\end{pmatrix}-\begin{pmatrix}2{,}000012 & 0\\ 0 & 2{,}000012\end{pmatrix}\right|=\left|\begin{pmatrix}0{,}007988 & 0\\ 0 & 0{,}007988\end{pmatrix}\right|$$

$0{,}007988<0{,}1$

**Example 2:**

Let $Z=x^4+2x^3y^2+y^2$ be a two-variable function. Using the graph of the function, we can choose the point $M(\frac{1}{2};0)$ that is close to the minimum value. We use the above formulas in sequence. We find the first and second derivatives of the function with respect to x and y:




$$\begin{cases}\dfrac{\partial z}{\partial x} = 4x^3 + 3x^2y^2 \\ \dfrac{\partial z}{\partial y} = 2x^3y + 2y\end{cases} \Rightarrow \frac{\partial^2 z}{\partial x^2} = 12x^2 + 6xy^2,\ \frac{\partial^2 z}{\partial x \partial y} = 6x^2y,\ \ \frac{\partial^2 z}{\partial y^2} = 2x^3 + 2$$

$$det(H) = \begin{pmatrix} 12x^2 + 6xy^2 & 6x^2y \\ 6x^2y & 2x^3 + 2 \end{pmatrix}$$

$$\mathrm{H}\left(\frac{1}{2};0\right) = \begin{pmatrix} 3 & 0 \\ 0 & \frac{9}{4} \end{pmatrix} = \frac{27}{4}$$

$$\det H^{-1}\left(\frac{1}{2};0\right) = \frac{4}{27}\begin{pmatrix} \frac{9}{4} & 0 \\ 0 & 3 \end{pmatrix}$$

$$\Delta f(x;y) = \begin{pmatrix} 4x^3 & +3x^2y^2 \\ 2x^3y & +2y \end{pmatrix}^T$$

$$\Delta f\left(\frac{1}{2};0\right) = \begin{pmatrix} \frac{1}{2} \\ 0 \end{pmatrix}$$

$$x_2 = \left(\frac{1}{2};0\right) - \frac{4}{27}\begin{pmatrix} \frac{9}{4} & 0 \\ 0 & 3 \end{pmatrix}\begin{pmatrix} \frac{1}{2} \\ 0 \end{pmatrix} = \left(\frac{1}{2};0\right) - \frac{4}{27}\left(\frac{9}{8};0\right) = \left(\frac{1}{2};0\right) - \left(\frac{1}{6};0\right) = \left(\frac{1}{3};0\right)$$

ISSN: 2835-5326 **Website:** econferenceseries.com

$$\left|H\left(\frac{1}{2};0\right)-H\left(\frac{1}{3};0\right)\right| = \left|\begin{pmatrix}3 & 0\\ 0 & \frac{9}{4}\end{pmatrix}-\begin{pmatrix}\frac{4}{3} & 0\\ 0 & \frac{91}{27}\end{pmatrix}\right| = \left|\begin{pmatrix}\frac{5}{3} & 0\\ 0 & \frac{38}{27}\end{pmatrix}\right| \quad \frac{5}{3}>0{,}1$$

$$\mathrm{H}\left(\frac{1}{3};0\right)=\begin{pmatrix}\frac{4}{3} & 0\\ 0 & \frac{91}{27}\end{pmatrix}=\frac{364}{81}$$

$$\Delta f\left(\frac{1}{3};0\right)=\begin{pmatrix}\frac{4}{27}\\ 0\end{pmatrix}$$

$$\det H^{-1}\left(\frac{1}{3};0\right)=\frac{81}{364}\begin{pmatrix}\frac{91}{27} & 0\\ 0 & \frac{4}{3}\end{pmatrix}$$

$$x_2=\left(\frac{1}{3};0\right)-\frac{81}{364}\begin{pmatrix}\frac{91}{27} & 0\\ 0 & \frac{4}{3}\end{pmatrix}\begin{pmatrix}\frac{4}{27}\\ 0\end{pmatrix}=\left(\frac{1}{3};0\right)-\frac{81}{364}\left(\frac{91*4}{27*27};0\right)\begin{pmatrix}\frac{4}{27}\\ 0\end{pmatrix}=$$

$$=\left(\frac{1}{3};0\right)-\left(\frac{1}{9};0\right)=\left(\frac{2}{9};0\right)$$

$$\left|H\left(\frac{1}{3};0\right)-H\left(\frac{2}{9};0\right)\right| = \left|\begin{pmatrix}\frac{4}{3} & 0\\ 0 & \frac{91}{27}\end{pmatrix}-\begin{pmatrix}\frac{84}{81} & 0\\ 0 & 2\frac{16}{729}\end{pmatrix}\right|=\left|\begin{pmatrix}\frac{24}{81} & 0\\ 0 & \frac{983}{729}\end{pmatrix}\right| \quad \frac{983}{729}>0.1$$

$$\mathrm{H}\left(\frac{2}{9};0\right)=\begin{pmatrix}\frac{84}{81} & 0\\ 0 & \frac{1474}{729}\end{pmatrix}=2{,}097$$

$$\Delta f\left(\frac{2}{9};0\right)=\begin{pmatrix}0{,}043\\ 0\end{pmatrix}$$

$$\det H^{-1}\left(\frac{2}{9};0\right)-\frac{1}{2{,}097}\begin{pmatrix}2{,}022 & 0\\ 0 & 1{,}037\end{pmatrix}$$

$$x_2=(0{,}222;0)-\frac{1}{2{,}097}\begin{pmatrix}2{,}022 & 0\\ 0 & 1{,}037\end{pmatrix}\begin{pmatrix}0{,}043\\ 0\end{pmatrix}=$$

$$=(0{,}222;0)-(0{,}041;0)=(0{,}181;0)$$

ISSN: 2835-5326 **Website:** econferenceseries.com

$$|H(0{,}222;0)-H(0{,}181;0)| = \left|\begin{pmatrix}1{,}037 & 0\\ 0 & 2{,}022\end{pmatrix} - \begin{pmatrix}0{,}393 & 0\\ 0 & 2{,}013\end{pmatrix}\right| = \left|\begin{pmatrix}0{,}643 & 0\\ 0 & 0{,}009\end{pmatrix}\right| \quad 0{,}643>0{,}1$$

$$H(0{,}181;0) = \begin{pmatrix}0{,}393 & 0\\ 0 & 2{,}013\end{pmatrix} = 0{,}791$$

$$\Delta f(0{,}181;0) = \begin{pmatrix}0{,}024\\ 0\end{pmatrix}$$

$$\det H^{-1}(0{,}181;0) - \frac{1}{0{,}791}\begin{pmatrix}2{,}013 & 0\\ 0 & 0{,}393\end{pmatrix}$$

$$x_2 = (0{,}181;0) - \frac{1}{0{,}791}\begin{pmatrix}2{,}013 & 0\\ 0 & 0{,}393\end{pmatrix}\begin{pmatrix}0{,}024\\ 0\end{pmatrix} =$$
$$= (0{,}181;0) - (0{,}061;0) = (0{,}12;0)$$

$$|H(0{,}181;0)-H(0{,}061;0)| = \left|\begin{pmatrix}0{,}393 & 0\\ 0 & 2{,}013\end{pmatrix} - \begin{pmatrix}0{,}303 & 0\\ 0 & 2{,}00045\end{pmatrix}\right| = \left|\begin{pmatrix}0{,}09 & 0\\ 0 & 0{,}0145\end{pmatrix}\right| \quad 0{,}09<0{,}1.$$

```
#file Launch.py
from Library3 import newton_method
from Library1 import objective,Gamma,x,y;
import sympy as sm
import numpy as np
newton_method(objective,Gamma,{x:0.5,y:0})
```

*The main file is located in the Launch.py program, this program is compiled based on 3 libraries.*

```
import sympy as sm
import numpy as np
# Define symbols & objective function
x, y = sm.symbols('x y')
Gamma = [x,y]
objective = x**4+2*x**3+y**2
```

```
def get_gradient(function, symbols):
    '''
    Helper function to solve for Gradient of SymPy function.
    '''
    d1 = {}
    gradient = np.array([])
    for s in symbols:
        d1[s]= sm.diff(function,s) # Take first derivative w/ respect to each symbol
        gradient = np.append(gradient, d1[s])
    return gradient
# Function to return the Hessian
def get_hessian(function, symbols):
    '''
    Helper function to solve for Hessian of SymPy function.
    '''
    d2 = {}
    hessian = np.array([])
    for s1 in symbols:
        for s2 in symbols:
            d2[f"{s1}{s2}"] = sm.diff(function,s1,s2) # Take second derivative w/
respect to each combination of symbols
            hessian = np.append(hessian, d2[f"{s1}{s2}"])
    hessian = np.array(np.array_split(hessian,len(symbols)))
    return hessian
```

*This Library1.py file is the first generated library that basically runs on 2 functions and returns a result*
*functions get_gradient() and get_hessian() have been created.*

```
#this file is Library3.py
import sympy as sm
import numpy as np
from Library2 import get_gradient,get_hessian
def newton_method(function,symbols,x0,iterations=100,mute=False):
    '''
```

```
Function to run Newton's method to optimize SymPy function.
'''
# Dictionary of values to record each iteration
x_star = {}
x_star[0] = np.array(list(x0.values()))

if not mute:
    print(f"Starting Values: {x_star[0]}")
i=0
while i < iterations:

    # Get gradient and hessian
    gradient = get_gradient(function, symbols, dict(zip(x0.keys(),x_star[i])))
    hessian = get_hessian(function, symbols, dict(zip(x0.keys(),x_star[i])))

    # Newton method iteration scheme
    x_star[i+1] = x_star[i].T - np.linalg.inv(hessian) @ gradient.T

    # Check convergence criteria
    if np.linalg.norm(x_star[i+1] - x_star[i]) < 10e-5:
        solution = dict(zip(x0.keys(),x_star[i+1]))
        print(f"\nConvergence Achieved ({i+1} iterations): Solution = {solution}")
        break
    else:
        solution = None

    if not mute:
        print(f"Step {i+1}: {x_star[i+1]}")

    i += 1

return solution
```

ISSN: 2835-5326 **Website:** econferenceseries.com

*The Result:*

```
PS D:\Gessian Matrix Program> & C:/Users/User/AppData/Local/Programs/Python/Python311/python.exe "d:/Gessian Matrix Program/Launch.py"
Starting Values: [0.5 0. ]
Step 1: [0.27777778 0.        ]
Step 2: [0.1489533 0.       ]
Step 3: [0.0776951 0.       ]
Step 4: [0.03978111 0.        ]
Step 5: [0.02014422 0.        ]
Step 6: [0.0101384 0.       ]
Step 7: [0.00508616 0.        ]
Step 8: [0.00254737 0.        ]
Step 9: [0.00127476 0.        ]
Step 10: [0.00063765 0.        ]
Step 11: [0.00031889 0.        ]
Step 12: [0.00015946 0.        ]

Convergence Achieved (13 iterations): Solution = {x: 7.973620423550828e-05, y: 0.0}
PS D:\Gessian Matrix Program>
```

*This is Library3.py file, the main result is sent in this library. This is newton_method(args,args,args..)*
*function returns the result.*
*Several problems arise in the application of Newton's method of solving optimization problems.*
*1) It is necessary to fulfill the condition that the* $H(x_k)$ *Hesse matrix is positive definite in each k-iteration, otherwise it may not be directed towards the minimum.*
*2)* $H(x_k)$ *- the matrix is an eigenmatrix and its inverse matrix may not exist.*
*3) One of the most important problems is to calculate the n x n matrix in each iteration and find its inverse, which requires a lot of calculations and time when the number of variables n is large.*
*4) Depending on the selection of the starting* $x_k$ *point, it may be necessary to perform multi-step calculations until the norm of the matrix is less than the given pression ε.*
*Therefore, there are various modifications of Newton's method to avoid or solve such problems.*

***Books:***

1. Lisa, D., Nicholas, M., An Introduction to Statistics and Data Analysis Using Stata SAGE 2019
2. Maurits, K ., Edwin ,H., Statistics for Data Scientists: An Introduction to Probability, Statistics, and Data Analysis Springer 2022.
3. FeruzaSaidovnaRakhimova in recognition of the paper publication of the research paper on Central Asian Journal of Mathematical Theory and Computer Science(CAJMTCS) with the title:ONTEACHINGSTUDENTSNEWTON'S

METHODOFSOLVING ONE-DIMENSIONAL OPTIMIZATION PROBLEMS Vol 4 No 1(2023):CAJMTCS

4. Ronald, L., Jonathan R., Multivariable Calculus with MATLAB : With Applications to Geometry and Physics Springer 2017 Ron, L., Bruce, E.,: (2005) Brief Calculus : An Applied Approach 7th edition, Cengage Learning
5. Rakhimov, Bakhtiyar; Rakhimova, Feroza; Sobirova, Sabokhat; Allaberganov, Odilbek; ,Mathematical Bases Of Parallel Algorithms For The Creation Of Medical Databases,InterConf, 2021.