Eur. J. Math. Anal. 5 (2025) 9
doi: 10.28924/ada/ma.5.9

Numerical Results for Gauss-Seidel Iterative Algorithm Based on Newton Methods for Unconstrained Optimization Problems

Nguyen Dinh Dung

TNU-University of Information and Communication Technology, Thai Nguyen, Vietnam
nddung@ictu.edu.vn

Abstract. Optimization problems play a crucial role in various fields such as economics, engineering, and computer science. They involve finding the best value (maximum or minimum) of an objective function. In unconstrained optimization problems, the goal is to find a point where the function's value reaches a maximum or minimum without being restricted by any conditions. Currently, there are many different methods to solve unconstrained optimization problems, one of which is the Newton method. This method is based on using a second-order Taylor series expansion to approximate the objective function. By calculating the first derivative (gradient) and second derivative (Hessian matrix) of the function, the Newton method determines the direction and step size to find the extrema. This method has a very fast convergence rate when near the solution and is particularly effective for problems with complex mathematical structures. In this paper, we introduce a Gauss-Seidel-type algorithm implemented for the Newton and Quasi-Newton methods, which is an efficient approach for finding solutions to optimization problems when the objective function is a convex functional. We also present some computational results for the algorithm to illustrate the convergence of the method.

1. Introduction

In this paper, we focus on solving the unconstrained nonlinear optimization problem

$$\min_{\mathbf{x}\in\mathbf{R}^n} f(\mathbf{x}) \tag{1}$$

where $f(\mathbf{x})$ is a convex functional with second derivative on $\mathbf{R}^n$. Optimization problem (1) is also a problem derived from many problems in different fields in economics and engineering. Solving problem (1) can lead to solving a system of nonlinear equations and has many different applications, for example in solving the ℓ_1-norm problem arising from compressing sensing [1]- [4], in variational inequalities problems [5]- [6], and optimal power flow equations [7] among others. In a broader sense, optimization should be understood as the activities aimed at obtaining the best result under certain conditions (maximizing profit, minimizing costs). The theory of optimization methods is not new, there are a huge number of optimization methods: methods based on the use

Received: 27 Oct 2024.

Key words and phrases. Convex optimization; Newton; Quasi-Newton; Hessen matrix; Gauss–Seidel.

of Lagrange multipliers, methods of dynamic programming, and methods of the calculus of variations, linear and nonlinear programming methods, there are currently many different solutions depending on the objective function. One of the simplest methods is the steepest descent method or also known as the Gradient descent method [2], [8], this method is simple and applicable to a fairly wide class of objective functions, the content of the method is to give a sequence of iterations $x^{(k+1)} = x^{(k)} - \alpha_k \nabla f(x_k)$, $\alpha_k > 0$, where α_k is the step length determined by Armijo's rule after the exact or inexact line search, this method has the disadvantage of linear convergence rate. We want to improve the efficiency of the algorithm's convergence, the Newton method is a good choice [9]- [12], Newton's method was first proposed by Isaac Newton in 1964 when finding solutions to nonlinear equations. To date, Newton's methods have widely been used for solving the unconstrained nonlinear optimization problem. As a result, studies of Newton's method form an extremely active area of research, with new variants being constantly developed and tested. Basic results on Newton's method and comprehensive lists of references can be found, e.g., in the books by Dennis and Schnabel [13], Ostrowski [14], Ortega and Rheinboldt [15], Deuflhard [16] and Corless and Fillion [17], survey of Newton's method in [18]. The general iterative rule for solving (1) starts from an initial approximation and generates a sequence using the general iterative scheme

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{d}_k. \tag{2}$$

Where $\mathbf{d}_k$ is an appropriate search direction. General class of algorithms of the form (2) is known as the line search algorithms. The method has a local quadratic convergence, thus converging extremely fast in a neighbourhood of the solution [19]. Nowadays, this method is extended to find optimal solutions for multivariable functions based on Taylor expansion, the solution of the optimization problem is performed in an iterative sequence $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \left[\nabla^2 f(\mathbf{x}^{(k)})\right]^{-1} \nabla f(\mathbf{x}^{(k)})$, if the objective function is not quadratic, the above iteration sequence may diverge or converge to a local minimum or converge to a saddle point. A variant of Newton's method introduced in [20] is the generalized Newton's method, in which the solution to the optimization problem is computed by the iteration sequence $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \alpha_k \left[\nabla^2 f(\mathbf{x}^{(k)})\right]^{-1} \nabla f(\mathbf{x}^{(k)})$, where α_k is called the step length and is determined by one-dimensional search methods in the direction $-\left[\nabla^2 f(\mathbf{x}^{(k)})\right]^{-1} \nabla f(\mathbf{x}^{(k)})$. However, in some practical problems when leading to optimization problems where the objective function does not have a second derivative, applying Newton's method is not feasible. To overcome this limitation, recently some results in [21] and [22] have proposed a quasi Newton algorithm for finding solutions to nonlinear optimization problems, showing the effectiveness of the method.

The numerical results in the papers all use Jacobi iteration, so we hope to improve the convergence of the algorithm by applying the idea of Gauss-Seidel iteration to the implementation of Newton and quansi-Newton methods. So, in this paper, we propose Gauss – Seidel algorithms implemented for the Newton and quasi-Newton method for finding solutions at each iteration step, in which we inherit the information of the component solutions calculated in the current

iteration instead of using the solutions calculated in the previous iteration. The computational results illustrating the algorithm are given to confirm the convergence of the algorithm. The paper is organized as follows. Section 2 presents the Newton's method and implementation of the Gauss-Seidel iterative algorithm based on Newton and Quasi-Newton methods. Experimental results illustrating the convergence are presented in Section 3. Finally, there are conclusions and references.

2. Proposed method

2.1. **Newton's method.**

Let x^* is the minimum point of the functional, then the necessary condition is $\frac{\partial f}{\partial x_i}(\mathbf{x}^*) = 0$. in order to determine the iterative sequence, we use the Taylor expansion for $f(\mathbf{x})$, we have

$$f(x) = f(\mathbf{x}^{(k)}) + \nabla f_k(\mathbf{x} - \mathbf{x}^{(k)}) + \frac{1}{2}(\mathbf{x} - \mathbf{x}^{(k)})^2 \mathbf{J}_k, \tag{3}$$

where $\nabla f(\mathbf{x}) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, ..., \frac{\partial f}{\partial x_n}\right)$, $\mathbf{J}_k$ is the Hessen matrix and is defined as

$$\nabla^2 f(x) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}.$$

Since 2, we have

$$\nabla f = \nabla f_k + (x - x^{(k)}) J_k. \tag{4}$$

So, we have the iterative process.

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - (\mathbf{J}_k)^{-1} \nabla f_k,\ k = 1, 2, 3, ..., n. \tag{5}$$

Formula (4) is called Newton's iteration formula with second-order convergence rate. The algorithm is implemented as follows:

Algorithm 1:

function $\mathbf{x}$=newton($\mathbf{x}^{(1)}$,ε);

k=1;

while($\|\nabla f_k\| > \varepsilon$)

$\mathbf{d}^{(k)} = -(\mathbf{J}_k)^{-1} \nabla f_k$;

$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{d}^{(k)}$;

k=k+1;

end;

$\mathbf{x} = \mathbf{x}^{(k)}$;

According to this algorithm, at the step k: $\mathbf{d}^{(k)} = -(\mathbf{J}_k)^{-1}\nabla f_k$, $x^{(k+1)} = x^{(k)} + d^{(k)}$, we implement component inheritance $\mathbf{x}_i^{(k+1)}$ calculated to calculate $x_j^{(k+1)}$, $j = i+1, ..., n$, So (5) is replaced by

$$x_i^{(k+1)} = x_i^{(k)} + d_i^{(k)}, \tag{6}$$

where

$$d_1^{(k)} = -\sum_{j=1}^{n}\left[(J_k)^{-1}\right]\nabla f(x_j^{(k)})$$

$$d_i^{(k)} = -\sum_{j=1}^{i-1}\left[(J_k)^{-1}\right]\nabla f(x_j^{(k+1)}) - \sum_{j=i+1}^{n}\left[(J_k)^{-1}\right]\nabla f(x_j^{(k)}),\ i = 2, ..., n$$

So, Newton's algorithm is updated as follows:

Algorithm 2:

function x=newton($\mathbf{x}^{(1)}$,ε);

k=*1*;

n=length($\mathbf{x}^{(0)}$);

while($\|\nabla f_k\| > \varepsilon$)

$\mathbf{d}^{(k)} = 0$;

for j=1:n

$d_1^{(k)} = d_1^{(k)} - \left[(\mathbf{J}_k)^{-1}\right]\nabla f(x_j^{(k)})$

end;

$x_1^{(k+1)} = x_1^{(k)} + d_1^{(k)}$

for i=2:n

for j=1:i-1

$d_i^{(k)} = d_i^{(k)} - \left[(\mathbf{J}_k)^{-1}\right]\nabla f(x_j^{(k+1)})$

end;

for j=i:n

$d_i^{(k)} = d_i^{(k)} - \left[(\mathbf{J}_k)^{-1}\right]\nabla f(x_j^{(k)})$

end;

$x_i^{(k+1)} = x_i^{(k)} + d_i^{(k)}$

end;

k=k+1;

end;

$\mathbf{x} = \mathbf{x}^{(k+1)}$

In case the objective function is not second-order differentiable, then instead of using Newton's method, we will use the quasi-Newtonian method.

2.2. **Quasi-Newton method.**

The idea of the quasi-Newton method [22] is derived from formula (5), we approximate the Hessen

matrix $\mathbf{J}_k$ by matrix $\mathbf{B}_k$ So, since (3), we have the following iteration process:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \alpha_k \left[\mathbf{B}_k\right]^{-1} \nabla f_k,\ k = 1..n, \tag{7}$$

where, α_k is the step length determined in the direction $S_k = -\left[\mathbf{B}_k\right]^{-1} \nabla f_k$, it can change at each iteration and satisfy the Wolfe condition [22].

$$\begin{aligned} &f(\mathbf{x}^{(k)} + \alpha_k S_k) \leq f(\mathbf{x}^{(k)}) + c_1 \alpha_k S_k^T \nabla f(\mathbf{x}^{(k)}) \\ &-S_k^T \nabla f(\mathbf{x}^{(k)} + \alpha_k S_k) \leq -c_2 S_k^T \nabla f(\mathbf{x}^{(k)}) \\ &0 < c_1 < c_2 < 1 \end{aligned} \tag{8}$$

α_k is determined by algorithm 3

Algorithm 3:

function α_k=LineSearch(f,x^k, S_k);

Initialize constants c_1, c_2, β satisfy $0 < c_1 < c_2 < 1; 0 < \beta < 1$

$\alpha = \alpha_0$

while($f(x^{(k)} + \alpha S_k) > f(x^{(k)}) + c_1 \alpha S_k^T \nabla f(x^{(k)})$ or $-S_k^T \nabla f(x^{(k)} + \alpha S_k) > -c_2 S_k^T \nabla f(x^{(k)})$)

$\alpha = \beta\alpha$;

end;

$\alpha_k = \alpha$;

In case the objective function is a convex function, the obtained optimal point is the global optimal solution of problem (1). It is easy to see in [13], if $\mathbf{B}_k = I$ then the iterative formula (7) is the steepest descent method published in [23]. B_k is an approximation matrix for the Hessen matrix and satisfies the condition

$$\nabla f(\mathbf{x}^{(k)}) = \nabla f(\mathbf{x}^{(k-1)}) - \mathbf{B}_k(\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}). \tag{9}$$

At $x^{(k+1)}$, we have

$$\nabla f(\mathbf{x}^{(k+1)}) = \nabla f(\mathbf{x}^{(k)}) - \mathbf{B}_{k+1}(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) \tag{10}$$

Or can write

$$\mathbf{B}_{k+1}\mathbf{d}_k = \mathbf{g}_k, \tag{11}$$

where,

$$d_k = x^{(k+1)} - x^{(k)},\ g_k = \nabla f_{k+1} - \nabla f_k$$

Formula (11) can be rewritten as follows:

$$\mathbf{d}_k = \left[\mathbf{B}_{k+1}\right]^{-1} \mathbf{g}_k, \tag{12}$$

where, $\mathbf{B}_{k+1}$ is a positive definite symmetric matrix and is updated according to the formula

$$\mathbf{B}_{k+1} = \mathbf{B}_k + CZZ^T. \tag{13}$$

Since (11), we have $\left(\mathbf{B}_k + \mathbf{C}\mathbf{Z}\mathbf{Z}^T\right)\mathbf{d}_k = \mathbf{g}_k$. So, $C\mathbf{Z} = \frac{\mathbf{g}_k - [\mathbf{B}_k]\mathbf{d}_k}{\mathbf{Z}^T\mathbf{d}_k}$. Let $\mathbf{Z} = \mathbf{g}_k - [\mathbf{B}_k]\,\mathbf{d}_k$, then $C = \frac{1}{\mathbf{Z}^T\mathbf{d}_k}$ and

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{(\mathbf{g}_k - \mathbf{B}_k\mathbf{d}_k)\,(\mathbf{g}_k - \mathbf{B}_k\mathbf{d}_k)^T}{(\mathbf{g}_k - \mathbf{B}_k\mathbf{d}_k)^T\,\mathbf{d}_k}. \tag{14}$$

We can also use the following calculation: $\mathbf{B}_{k+1} = \mathbf{B}_k + C_1\mathbf{Z}_1\mathbf{Z}_1^T + C_2\mathbf{Z}_2\mathbf{Z}_2^T$, $\mathbf{d}_k = \mathbf{B}_k\mathbf{g}_k + C_1\mathbf{Z}_1(\mathbf{Z}_1^T\mathbf{g}_k) + C_2\mathbf{Z}_2(\mathbf{Z}_2^T\mathbf{g}_k)$. Let $\mathbf{Z}_1 = \mathbf{d}_k$ and $\mathbf{Z}_2 = \mathbf{B}_k\mathbf{g}_k$, similar to formula (15), we have

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{\mathbf{g}_k\mathbf{g}_k^T}{\mathbf{g}_k^T\mathbf{d}_k} - \frac{(\mathbf{B}_k\mathbf{d}_k)\,(\mathbf{B}_k\mathbf{d}_k)^T}{\mathbf{d}_k^T\mathbf{B}_k\mathbf{d}_k} \tag{15}$$

Thus, the algorithm to find the solution of problem (1) is implemented as follows:

Algorithm 4:

function x=QNewton($\mathbf{x}^{(1)}$, ε);

k=1

$\mathbf{B}_k = I$;

while($\|\nabla f_k\| > \varepsilon$)

$\mathbf{S}_k = -[\mathbf{B}_k]^{-1}\nabla f_k$;

α_k= LineSearch($f, x^{(k)}, \mathbf{S}_k$);

$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k\mathbf{S}_k$;

$\mathbf{d}_k = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}$;

$\mathbf{g}_k = \nabla f_{k+1} - \nabla f_k$;

Update $\mathbf{B}_{k+1}$ according to (14) or (15);

k=k+1;

end;

$\mathbf{x} = \mathbf{x}^{(k+1)}$;

According to this algorithm, starting from point $\mathbf{x}^{(1)}$, The iteration sequence (15) converges to the local optimum x^* and satisfied.

$$f(\mathbf{x}^{(1)}) \geq \ldots \geq f(\mathbf{x}^{(k)}) \geq \ldots \geq f(\mathbf{x}^*)$$

In case the objective function is a convex function, the obtained optimal point is the global optimal solution of problem (1). To illustrate the theoretical results, here are some experimental calculation results of the algorithm.

3. Experimental results and discussions

In this section, we perform experimental calculations to illustrate the convergence of the algorithm introduced in the paper. The data is given:

Objective function

$$f(x) = \sum_{i=1}^{10}(x_i - i)^4 \tag{16}$$

Initial approximation: $\mathbf{x}^{(1)} = (0, 0, ..., 0)$. It is easy to see that the exact solution of problem (1) is $\mathbf{x}^* = (1, 2, ..., 10)$, $f(\mathbf{x}^*) = 0$. The objective function (16) is differentiable at all levels, so the Hessen matrix exists, so we can completely apply Newton's algorithm to solve problem (1). Let $err = \left\|\mathbf{x}^{(k)} - \mathbf{x}^*\right\|_2$, we have the computational results illustrating the convergence of Newton's algorithm given in Table 1:

Table 1. *Approximate solution of problem (1) obtained from Algorithm 2*

$\mathbf{x}^{(k)}$	$k = 5$	$k = 10$	$k = 15$	$k = 20$
$x_1^{(k)}$	0.8025	0.9740	0.9966	0.9995
$x_2^{(k)}$	1.6049	1.9480	1.9931	1.9991
$x_3^{(k)}$	2.4074	2.9220	2.9897	2.9986
$x_4^{(k)}$	3.2099	3.8960	3.9863	3.9982
$x_5^{(k)}$	4.0123	4.8699	4.9829	4.9977
$x_6^{(k)}$	4.8148	5.8439	5.9794	5.9973
$x_7^{(k)}$	5.6173	6.8179	6.9760	6.9968
$x_8^{(k)}$	6.4198	7.7919	7.9726	7.9964
$x_9^{(k)}$	7.2222	8.7659	8.9692	8.9959
$x_{10}^{(k)}$	8.0247	9.7399	9.9657	9.9955
err	3.8758	0.5104	0.0672	0.0089

The calculation results in Table 1 show that the approximate solution found converges to the exact solution of the problem according to the number of iterations. The graphs in Figure 1 and Figure 2 illustrate the convergence of the algorithm.

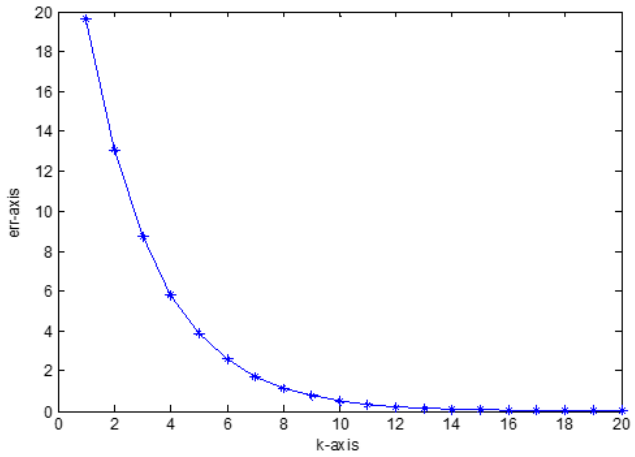

Figure 1. *Error graph according to the number of iterations with the number of iterations k=1,2,...,20 obtained from Algorithm 2*

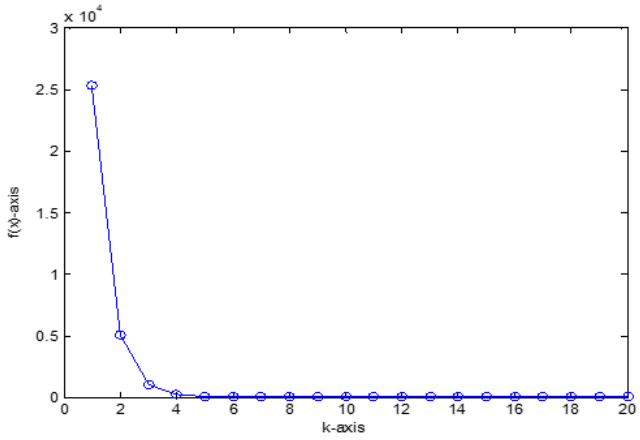

Figure 2. *Objective function graph according to the number of iterations (number of iterations k=1,2,...,20) obtained from Algorithm 2*

From Figure 1 and Figure 2, it can be seen that the error function and the objective function are monotonically decreasing functions with the number of iterations, which shows that the approximate solution converges to the exact solution of Problem (1). Now, let us consider the following objective function:

$$f(x) = \begin{cases} \sum_{i=1}^{10} \left(x_i - \frac{1}{i}\right)^4 & \exists x_i < \frac{1}{i} \\ \sum_{i=1}^{10} \left(x_i - \frac{1}{i}\right) \sqrt{x_i - \frac{1}{i}} & \forall x_i \geq \frac{1}{i} \end{cases} \tag{17}$$

The objective function does not have a second derivative at $x^* = \left(1, \frac{1}{2}, \ldots, \frac{1}{10}\right)$, So, in order to find the solution for Problem (1) with objective function (16), we perform Algorithm 4 with the Hessen matrix approximation. Matrix updated according to formula (15). The calculation results are given in Table 2.

Table 2. *Approximate solution of problem (1) obtained from Algorithm 4*

$\mathbf{x}^{(k)}$	$k = 5$	$k = 10$	$k = 15$	$k = 20$
$x_1^{(k)}$	0.4598	0.8654	0.9708	1.0128
$x_2^{(k)}$	0.5080	0.5079	0.5078	0.5078
$x_3^{(k)}$	0.2322	0.3231	0.3459	0.3548
$x_4^{(k)}$	0.1490	0.2424	0.2659	0.2748
$x_5^{(k)}$	0.0987	0.1791	0.1997	0.2078
$x_6^{(k)}$	0.0672	0.1341	0.1525	0.1600
$x_7^{(k)}$	0.0471	0.1025	0.1196	0.1271
$x_8^{(k)}$	0.0340	0.0798	0.0959	0.1039
$x_9^{(k)}$	0.0251	0.0632	0.0784	0.0866
$x_{10}^{(k)}$	0.0190	0.0508	0.0649	0.0733
err	0.6032	0.1680	0.0722	0.0584

The calculation results in Table 2 show that the approximate solution found converges to the exact solution of the problem depending on the number of iterations. The calculation results show that the quasi-Newton method has the advantage of not requiring a quadratic differentiable objective function, but the convergence is quite slow compared to the Newton method. The error function and the objective function are not monotonically decreasing functions with the number of iterations, but tend to decrease gradually, which also confirms that the approximate solution converges to the exact solution of the problem. The graphs in Figure 3 and Figure 4 illustrate the convergence of the algorithm.

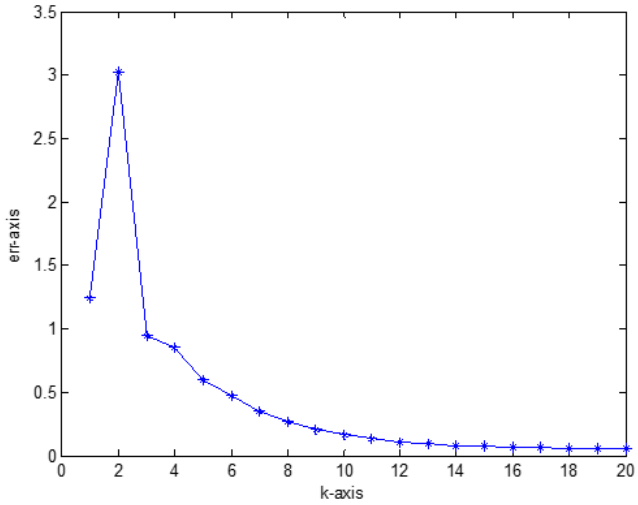

Figure 3. *Error graph depending on the number of iterations with the number of iterations k=1,2,...,20 obtained from Algorithm 4*

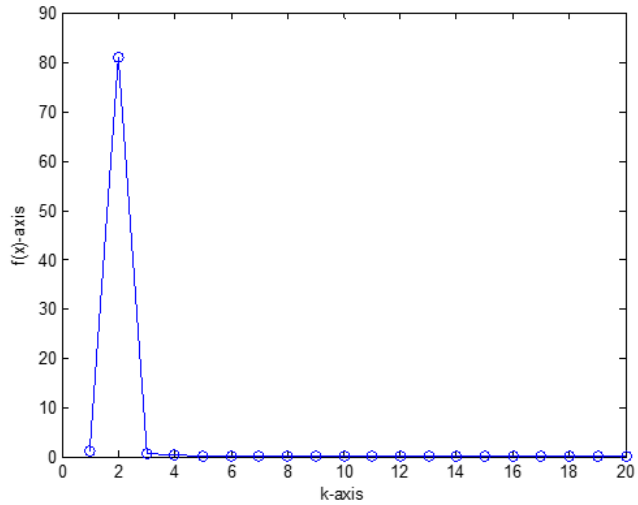

Figure 4. *Graph of the objective function depending on the number of iterations (number of iterations k=1,2,...,20) obtained from Algorithm 4*

4. Conclusion

In this paper, we implement an iterative algorithm to solve the unconstrained convex optimization problem based on Newton and Quasi-Newton iteration methods, in which the information of the component solutions calculated in the current iteration is inherited instead of using the solutions calculated in the previous iteration. The computational results according to the algorithm are performed on the Matlab 2014 environment, the numerical results have confirmed the convergence of the method and are consistent with the theory presented in the paper.

References

[1] S. Aji, P. Kumam, A.M. Awwal, M.M. Yahaya, W. Kumam, Two Hybrid Spectral Methods With Inertial Effect for Solving System of Nonlinear Monotone Equations With Application in Robotics, IEEE Access 9 (2021) 30918–30928. https://doi.org/10.1109/ACCESS.2021.3056567.

[2] Y. Zhou, Y. Wu, X. Li, A New Hybrid PRPFR Conjugate Gradient Method for Solving Nonlinear Monotone Equations and Image Restoration Problems, Math. Probl. Eng. 2020 (2020) 6391321. https://doi.org/10.1155/2020/6391321.

[3] M. Eshaghnezhad, S. Effati, A. Mansoori, A Neurodynamic Model to Solve Nonlinear Pseudo-Monotone Projection Equation and Its Applications, IEEE Trans. Cybern. 47 (2017) 3050–3062. https://doi.org/10.1109/TCYB.2016.2611529.

[4] S. Crisci, M. Piana, V. Ruggiero, M. Scussolini, A Regularized Affine-Scaling Trust-Region Method for Parametric Imaging of Dynamic PET Data, SIAM J. Imaging Sci. 14 (2021) 418–439. https://doi.org/10.1137/20M1336370.

[5] J.K. Liu, X.L. Du, M. Scussolini, A Gradient Projection Method for the Sparse Signal Reconstruction in Compressive Sensing, Appl. Anal. 97 (2018) 2122–2131. https://doi.org/10.1080/00036811.2017.1400510.

[6] A.M. Awwal, L. Wang, P. Kumam, H. Mohammad, W. Watthayu, A Projection Hestenes–stiefel Method With Spectral Parameter for Nonlinear Monotone Equations and Signal Processing, Math. Comput. Appl. 25 (2020) 27. https://doi.org/10.3390/mca25020027.

[7] B. Ghaddar, J. Marecek, M. Mevissen, Optimal Power Flow as a Polynomial Optimization Problem, IEEE Trans. Power Syst. 31 (2016) 539–546. https://doi.org/10.1109/TPWRS.2015.2390037.

[8] H.B. Curry, The Method of Steepest Descent for Non-Linear Minimization Problems, Quart. Appl. Math. 2 (1944) 258–261. https://doi.org/10.1090/qam/10667.

[9] D. Kovalev, K. Mishchenko, P. Richtárik, Stochastic Newton and Cubic Newton Methods With Simple Local Linear-Quadratic Rates, arXiv preprint arXiv:1912.01597 (2019). https://arxiv.org/abs/1912.01597.

[10] R. Weerakoon, T.G.I. Fernando, A New Variant of Newton's Method Based on the Rectangular Rule for Solving Nonlinear Equations, Appl. Math. Lett. 13 (2000) 87–93. https://doi.org/10.1016/S0893-9659(00)00112-1.

[11] A. Forsgren, P.E. Gill, M.H. Wright, Interior Methods for Nonlinear Optimization, SIAM Rev. 44 (2002) 525–597. https://doi.org/10.1137/S0036144502414942.

[12] R. Fletcher, Practical Methods of Optimization, Second Edition, John Wiley and Sons, Chichester, (2000), 40-46.

[13] J.E. Dennis Jr., R.B. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, SIAM Classics in Applied Mathematics, SIAM, Philadelphia (1996). https://doi.org/10.1137/1.9781611971200.

[14] A.M. Ostrowski, Solution of Equations and Systems of Equations, Academic Press, Basel (1960).

[15] J.M. Ortega, W.C. Rheinboldt, Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York (1970).

[16] P. Deuflhard, Newton Methods for Nonlinear Problems: Affine Invariance and Adaptive Algorithms, 2nd edition, Springer, Berlin (2011). https://doi.org/10.1007/978-3-642-23899-4.

[17] R.M. Corless, N. Fillion, A Graduate Introduction to Numerical Methods: From the Viewpoint of Backward Error Analysis, Springer, New York (2013). https://doi.org/10.1007/978-1-4614-8453-0.

[18] B.T. Polyak, Newton's Method and Its Use in Optimization, Eur. J. Oper. Res. 181 (2007) 1086–1096. https://doi.org/10.1016/j.ejor.2005.06.076.

[19] L.V. Kantorovich, Functional Analysis and Applied Mathematics, Uspekhi Mat. Nauk 3 (1948) 89–185. https://doi.org/10.1070/RM1948v003n06ABEH003995.

[20] K. Geleta, R.K. Vasudeva, Quasi-Newton Line Search Algorithm for Solving Unconstrained Non-Linear Least Square Optimization Problem, Int. J. Basic Appl. Sci. 10 (2021), 11-21.

[21] I. Povalej, Quasi-Newton's Method for Multiobjective Optimization, J. Comput. Appl. Math. 255 (2014) 765–777. https://doi.org/10.1016/j.cam.2013.06.045.

[22] K. Geleta, R.K. Vasudeva, Quasi-Newton Line Search Algorithm for Solving Unconstrained Non-Linear Least Square Optimization Problem, Int. J. Basic Appl. Sci. 13 (2021) 9–18.

[23] G. Yuan, S. Lu, Z. Wei, A Line Search Algorithm for Unconstrained Optimization, J. Softw. Eng. Appl. 3 (2010) 503–509. https://doi.org/10.4236/jsea.2010.35057.