Journal of Applied Mathematics and Computing (2020) 62:37–53
https://doi.org/10.1007/s12190-019-01272-z

ORIGINAL RESEARCH

A two-step improved Newton method to solve convex unconstrained optimization problems

T. Dehghan Niri[1] · **S. A. Shahzadeh Fazeli**[1] · **M. Heydari**[1]

Received: 12 May 2019 / Published online: 5 July 2019

Abstract
In this investigation, a new two-step Newton method to solve convex unconstrained optimization problems is developed. This proposed method is based on Traub's iterative scheme (Iterative methods for the solution of equations, Prentice Hall, Englewood Cliffs, 1964) which is extended to n-variable. The presented two-step algorithm is a modification of Newton method for solving unconstrained optimization problems. The convergence analysis for this iterative algorithm is established under suitable conditions. Various numerical examples are given to illustrate the efficiency and performance of the newly suggested method.

Keywords Newton method · Unconstrained optimization problems · Convergence analysis

1 Introduction

The general form of optimization problems can be written as follows:

$$\begin{aligned} \min \quad & f(x), \\ s.t. \quad & c_i(x) = 0,\ i \in E, \\ & c_j(x) \geq 0,\ j \in I, \\ & x \in \mathbb{R}^n, \end{aligned} \tag{1}$$

✉ S. A. Shahzadeh Fazeli
fazeli@yazd.ac.ir

T. Dehghan Niri
T.dehghan@stu.yazd.ac.ir

M. Heydari
m.heydari@yazd.ac.ir

[1] Department of Mathematics, Yazd University, Yazd, Iran

where E and I are, respectively, the index set of equality and inequality constraints, $c_i(x)$, $(i = 1, \ldots, m \in E \cup I)$ are constraint functions. When both objective function and constraint functions are linear functions, the problem is called linear programming. Otherwise, the problem is called nonlinear programming. Also, a problem that does not entail any equality or inequality constraints is said to be an unconstrained optimization problem. Now, we consider an unconstrained optimization problem

$$\min_{x \in \mathbb{R}^n} f(x), \tag{2}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a smooth and continuously differentiable function. It is not easy to find a global minimizer of $f(x)$, because our knowledge of the objective function is commonly only local. Therefore most algorithms are able to find only a local minimizer, which is a point that achieves the smallest value of $f(x)$ in its neighborhood [2]. In other words, we say that a point x^* is a local minimizer if there is a neighborhood $\mathcal{N}$ (an open set that contains x^*) of x^* such that $f(x^*) \leq f(x)$ for all $x \in \mathcal{N}$. Most of the numerical methods for unconstrained optimization problems can be classified into two groups, line search algorithms and trust region algorithms. There are many useful algorithms for solving the problem (2) such as the conjugate gradient methods, the trust region methods, the quasi-Newton methods, the classical Newton method, the Nelder–Meade simplex method for problems with noisy functions, the Levenberg–Marquardt method and etc. [2–4]. Among the methods mentioned above, the classical Newton method is very famous for its fast convergence property. There are several modifications of the Newton method for unconstrained minimization to achieve global and local convergence, see [2,4] and the references therein. In Newton method, the positive definiteness of the Hessian matrix of the objective function is an essential condition to get the local minimum and the fast local convergence. Zhou et al. [5] introduced a new algorithm for monotone equations and showed its superliner convergence under a local error-bound assumption that is weaker than the standard nonsingularity condition. A new trust region method for nonlinear equations with the trust region radius converging to zero is proposed in [6], and its convergence under some weak conditions is provided. Li et al. [7] obtained two regularized Newton methods for convex minimization problems in which the Hessian at solutions may be singular and showed that if the objective function be in LC^2, then the methods possess local quadratic convergence under a local error bound condition without the requirement of isolated nonsingular solutions. Zhou and Chen in [8] proposed a modified regularized Newton method for convex minimization problems whose Hessian matrices may be singular. Nesterov and Polyak [9] proposed a cubic regularization of the Newton method. At each iteration, it requires solving an unconstrained minimization problem. Nesterov and Nemirovsky [10] presented the class of self-concordant functions that is three time differentiable convex function with the second and third derivatives satisfying a particular condition at each point. Polyak [11] introduced the regularized Newton method for unconstrained convex optimization. For any convex function, with a bounded optimal set, the RNM generates a sequence that converges to the optimal set from any starting point. Dehghan et al. [12] inroduced a new modification of the Newton method to solve unconstrained optimization problems. Also,

they used a new improved Newton method in trust region algorithm for unconstrained minimization problems and analyzed its local and global convergence [13]. Recently, Dehghan et al. [14] proposed a new regularized Newton method based on Q.I.F method to solve optimization problems. One of the applications of the above algorithms is to solve system of nonlinear equations. For example, the system of nonlinear equations appearing in the references [15–19] can be solved by using these methods.

In this paper, we introduce a new algorithm to solve the convex unconstrained optimization problems. The organization of the paper is as follows: In Sect. 2, a new algorithm for solving unconstrained minimization problems is presented. Its associated convergence analysis is given in Sect. 3. Some numerical results to compare the new proposed method with the other algorithms are reported in Sect. 4 and finally the conclusions are described in Sect. 5.

2 Description of the method

In this section, a brief review of Newton method for unconstrained optimization problems is given which mentioned in [2]. Consider the minimization problem,

$$\min_{x\in\mathbb{R}^n} f(x), \tag{3}$$

where $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ is a smooth function and twice continuously differentiable. Gradient $\nabla f(x)$ and Hessian matrix $\nabla^2 f(x)$ are denoted by $g(x)$ and $H(x)$, respectively. In the line search method, each iteration computes a search direction p_k and then decides how far to move along that direction. Iterations are as follows:

$$x_{k+1} = x_k + \alpha_k p_k, \tag{4}$$

where the positive scalar α_k is called the step length. The success of linear search method depends on the appropriate selections of step length α_k and direction p_k. Most of the line search methods require p_k to be a descent direction, because this property guarantees that the function $f(x)$ can be reduced along this direction. The search direction can be defined as:

$$p_k = -B_k^{-1}\nabla f_k, \tag{5}$$

where B_k is a symmetric and nonsingular matrix.

The Newton method is a powerful technique for solving nonlinear equations. It is an application of Taylor Polynomials for finding roots of functions. Newton method is the iterative method for finding a simple root x^* of nonlinear equation $f(x)$, i.e., $f(x^*) = 0,\ f'(x^*) \neq 0$, by using

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \quad k = 0, 1, \ldots, \tag{6}$$

that converges quadratically in some neighborhood of x^* [1,20–22]. Newton method can also be used to find a minimum or maximum of a function. The derivative is zero at a minimum or maximum, so minima and maxima can be found by applying Newton method to the derivative. The iteration becomes:

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}, \quad k = 0, 1, \ldots. \tag{7}$$

The above method can be generalized to several dimensions by replacing the derivative with the gradient, $\nabla f(x_k)$, and the reciprocal of the second derivative with the inverse of the Hessian matrix, $\nabla^2 f(x_k)$. So, we have the following iterative formula

$$x_{k+1} = x_k - \left(\nabla^2 f(x_k)\right)^{-1} \nabla f(x_k), \quad k = 0, 1, \ldots. \tag{8}$$

In the line search Newton method, B_k is the Hessian matrix $\nabla^2 f(x_k)$. If the Hessian matrix is not positive definite, or is close to being singular, then we can modify this matrix before or during the solution process. Following is a general description of this method.

Algorithm 1. (Line Search Newton with Modification) [2]:

For given initial point x_0 and parameters $\alpha_0 > 0, \beta > 0$;
while $\nabla f_k \neq 0$
Factorize the matrix $B_k = \nabla^2 f_k + E_k$, where $E_k = 0$ if $\nabla^2 f_k$ is sufficiently positive definite; otherwise, E_k is chosen to ensure that B_k is sufficiently positive definite.
Solve $B_k p_k = -\nabla f_k$;
Set $x_{k+1} = x_k + \alpha_k p_k$,
where α_k satisfies the Wolfe, Goldstein, or Armijo backtracking conditions.
end while.

The choice of Hessian modification E_k is crucial to the effectiveness of the method. The modified Newton method for multiple root x^* of multiplicity m, i.e., $f^{(j)}(x^*) = 0,\ j = 0, 1, \ldots, m-1$ and $f^{(m)}(x^*) \neq 0$, is quadratically convergent and it is written as:

$$x_{k+1} = x_k - m\frac{f(x_k)}{f'(x_k)}, \quad k = 0, 1, \ldots. \tag{9}$$

If the multiplicity m is unknown, the standard Newton method has a linear convergence with a rate of $\frac{(m-1)}{m}$ [23]. Traub [1] used a transformation $u(x) = \frac{f(x)}{f'(x)}$ instead of $f(x)$ for computing a multiple root of $f(x) = 0$. Then the problem of finding a multiple root is reduced to the problem of finding a simple root of the transformed equation $u(x)$, and thus any iterative method can be used preserving its original convergence order. Applying the standard Newton method (6) to $u(x) = 0$, we can obtain

$$x_{k+1} = x_k - \frac{f(x_k) f'(x_k)}{f'(x_k)^2 - f(x_k) f''(x_k)}, \quad k = 0, 1, \ldots. \tag{10}$$

This method can be extended to n-variable functions ($f : \mathbb{R}^n \to \mathbb{R}$) as

$$x_{k+1} = x_k - \left(\nabla f(x_k) \nabla f(x_k)^T - f(x_k) \nabla^2 f(x_k) \right)^{-1} f(x_k) \nabla f(x_k), \quad k = 0, 1, \ldots. \tag{11}$$

Now, we propose a two-step algorithm to solve unconstrained optimization problems by using methods (8) and (11).

Algorithm 2. (Proposed method)

Step 1. Given an initial point x_0, $\tau > 0$, $\epsilon > 0$ and $\theta \in \mathbb{R}$.
Step 2. If $\|g_k\| \leq \epsilon$ stop, else go to Step 3.
Step 3. If $\left(g_k g_k^T - f_k H_k\right)$ is a positive definite matrix then
Solve $\left(g_k g_k^T - f_k H_k\right) p_k = -f_k g_k$;
else
Solve $\left(g_k g_k^T - f_k H_k + \tau I\right) p_k = -f_k g_k$;
Step 5. If H_k is sufficiently positive definite set $B_k = H_k$;
otherwise, E_k is chosen to ensure that B_k is sufficiently positive definite,
Solve $B_k \tilde{p}_k = -g_k$;
Step 6. $x_{k+1} = x_k + \theta p_k + (1 - \theta) \tilde{p}_k$.
set $k := k + 1$ and go to Step 2.

It is clear that, the introduced matrix $(g_k g_k^T - f_k H_k)$ is a symmetric matrix. Furthermore, in this algorithm, there is no need to calculate the step length and $\alpha_k = 1$ at each iteration.

3 Convergence analysis

In this section, we study convergence of the proposed method. The following assumptions are imposed throughout the paper.

Assumption 3.1

(A1): Let $f : \Lambda \subset \mathbb{R}^n \to \mathbb{R}$ be defined on the bounded and close set Λ. Suppose f is twice continuously differentiable and let $\overline{\Lambda}(x_0)$ denote the closure of the level set,

$$\Lambda(x_0) = \{x : x \in \Lambda, f(x) \leq f(x_0)\}. \tag{12}$$

(A2): $\nabla f(x)$ and $\nabla^2 f(x)$ are both Lipschitz continuous that is, there exists constants $L_1 > 0$ and $L_2 > 0$ such that

$$\|\nabla f(x) - \nabla f(y)\| \leq L_1 \|x - y\|, \quad x, y \in \mathbb{R}^n, \tag{13}$$

and

$$\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq L_2 \|x - y\|, \quad x, y \in \mathbb{R}^n. \tag{14}$$

(A3): $2(\nabla f_k^T (f_k \nabla^2 f_k)^{-1} \nabla f_k) \leq 1$.
(A4): $L = \max\{L_1, L_2\}$ and $\gamma = L\|(\nabla^2 f(x^*))^{-1}\| < 1$.
(A5): $\theta \in \mathbb{R}$ and $|\theta| < \frac{1-\gamma}{6\gamma}$.

Theorem 3.2 *Suppose A is a nonsingular $N \times N$ matrix, U is $N \times M$, V is $M \times N$, then $A + UV$ is nonsingular if and only if $I + VA^{-1}U$ is a nonsingular $M \times M$ matrix. If this is the case, then*

$$(A + UV)^{-1} = A^{-1} - A^{-1}U(I + VA^{-1}U)^{-1}VA^{-1}. \tag{15}$$

This is the Sherman–Morrison–Woodbury formula [3,24,25]. See [3] for further generalizations.

Proposition 3.3 [3] *Let B be a nonsingular $n \times n$ matrix and let $u, v \in \mathbb{R}^n$. Then $B + uv^T$ is invertible if and only if $1 + v^T B^{-1} u \neq 0$. In this case,*

$$(B + uv^T)^{-1} = \left(I - \frac{B^{-1}uv^T}{1 + v^T B^{-1} u}\right) B^{-1}. \tag{16}$$

Lemma 3.4 *Suppose that Assumption 3.1 **(A1)** and **(A3)** hold. Then*

(I) $\left|\frac{\nabla f_k^T (-f_k \nabla^2 f_k)^{-1} \nabla f_k}{1 + \nabla f_k^T (-f_k \nabla^2 f_k)^{-1} \nabla f_k}\right| \leq 1$.

(II) $(-f_k \nabla^2 f_k + \nabla f_k \nabla f_k^T)^{-1} = (-f_k \nabla^2 f_k)^{-1} \left(I - \frac{\nabla f_k \nabla f_k^T (-f_k \nabla^2 f_k)^{-1}}{1 + \nabla f_k^T (-f_k \nabla^2 f_k)^{-1} \nabla f_k}\right)$.

Proof From Assumption 3.1 **(A3)**, we have

$$2(\nabla f_k^T (f_k \nabla^2 f_k)^{-1} \nabla f_k) \leq 1 \Longrightarrow (\nabla f_k^T (-f_k \nabla^2 f_k)^{-1} \nabla f_k) \geq -\frac{1}{2}, \tag{17}$$

and hence

$$\left|\frac{\nabla f_k^T (-f_k \nabla^2 f_k)^{-1} \nabla f_k}{1 + \nabla f_k^T (-f_k \nabla^2 f_k)^{-1} \nabla f_k}\right| \leq 1. \tag{18}$$

According to Theorem (3.2) and Proposition (3.3), we set $B = -f_k \nabla^2 f_k$, $u = v = \nabla f_k$. From (17) we obtain

$$1 + v^T B^{-1} u = 1 + \nabla f_k^T (-f_k \nabla^2 f_k)^{-1} \nabla f_k \geq \frac{1}{2}. \tag{19}$$

Therefore, the matrix $B + uv^T$ is invertible and we can get

$$\begin{aligned}(B+uv^T)^{-1} &= (-f_k\nabla^2 f_k + \nabla f_k \nabla f_k^T)^{-1} \\ &= (-f_k\nabla^2 f_k)^{-1} - (-f_k\nabla^2 f_k)^{-1}\nabla f_k(1+\nabla f_k^T \\ &\quad (-f_k\nabla^2 f_k)^{-1}\nabla f_k)^{-1}\nabla f_k^T(-f_k\nabla^2 f_k)^{-1} \\ &= \frac{-1}{f_k}(\nabla^2 f_k)^{-1}\left(I - \frac{\nabla f_k \nabla f_k^T(-f_k\nabla^2 f_k)^{-1}}{1+\nabla f_k^T(-f_k\nabla^2 f_k)^{-1}\nabla f_k}\right). \end{aligned} \tag{20}$$

□

Theorem 3.5 *Suppose that Assumption 3.1* ***(A1)–(A5)*** *hold. Assume that*

$$\left(\nabla f(x_k)\nabla f(x_k)^T - f(x_k)\nabla^2 f(x_k)\right) p_k = -f(x_k)\nabla f(x_k), \tag{21}$$

and

$$\nabla^2 f(x_k)\tilde{p}_k = -\nabla f(x_k). \tag{22}$$

Then the iteration $x_{k+1} = x_k + \theta p_k + (1-\theta)\tilde{p}_k$ generated by Algorithm 2 converges to x^*, where x_0 is the starting point.

Proof In this proof, $f_k, \nabla f_k$ and $\nabla^2 f_k$ denotes the $f(x_k), \nabla f(x_k)$ and $\nabla^2 f(x_k)$, respectively. By using Lemma 3.4-**II**

$$p_k = (\nabla^2 f_k)^{-1}\left(I - \frac{\nabla f_k \nabla f_k^T(-f_k\nabla^2 f_k)^{-1}}{1+\nabla f_k^T(-f_k\nabla^2 f_k)^{-1}\nabla f_k}\right)\nabla f_k. \tag{23}$$

Without loss of generality, we assume $\nabla^2 f_k$ positive definite matrix. According to (22) and (23) we have

$$\begin{aligned} & x_{k+1} - x^* \\ &= x_k + \theta p_k + (1-\theta)\tilde{p}_k - x^* \\ &= x_k - x^* + \theta(\nabla^2 f_k)^{-1}\left(I - \frac{\nabla f_k \nabla f_k^T(-f_k\nabla^2 f_k)^{-1}}{1+\nabla f_k^T(-f_k\nabla^2 f_k)^{-1}\nabla f_k}\right)\nabla f_k - (1-\theta)(\nabla^2 f_k)^{-1}\nabla f_k \\ &= (\nabla^2 f_k)^{-1}\left(\nabla^2 f_k(x_k - x^*) + \theta\left(I - \frac{\nabla f_k \nabla f_k^T(-f_k\nabla^2 f_k)^{-1}}{1+\nabla f_k^T(-f_k\nabla^2 f_k)^{-1}\nabla f_k}\right)\nabla f_k - (1-\theta)\nabla f_k\right) \\ &= (\nabla^2 f_k)^{-1}\left(\nabla^2 f_k(x_k - x^*) + \theta\nabla f_k - \theta\frac{\nabla f_k \nabla f_k^T(-f_k\nabla^2 f_k)^{-1}\nabla f_k}{1+\nabla f_k^T(-f_k\nabla^2 f_k)^{-1}\nabla f_k} - (1-\theta)\nabla f_k\right) \\ &= (\nabla^2 f_k)^{-1}\left(\nabla^2 f_k(x_k - x^*) + 2\theta\nabla f_k - \nabla f_k - \theta\frac{\nabla f_k \nabla f_k^T(-f_k\nabla^2 f_k)^{-1}\nabla f_k}{1+\nabla f_k^T(-f_k\nabla^2 f_k)^{-1}\nabla f_k}\right), \end{aligned} \tag{24}$$

and hence

$$\begin{aligned}
&\|x_{k+1}-x^*\| \\
&= \left\| (\nabla^2 f_k)^{-1} \left(\nabla^2 f_k (x_k - x^*) + 2\theta \nabla f_k - \nabla f_k - \theta \frac{\nabla f_k \nabla f_k^T (-f_k \nabla^2 f_k)^{-1} \nabla f_k}{1 + \nabla f_k^T (-f_k \nabla^2 f_k)^{-1} \nabla f_k} \right) \right\| \\
&\leq \|(\nabla^2 f_k)^{-1}\| \left\| \nabla^2 f_k (x_k - x^*) + 2\theta \nabla f_k - \nabla f_k - \theta \frac{\nabla f_k \nabla f_k^T (-f_k \nabla^2 f_k)^{-1} \nabla f_k}{1 + \nabla f_k^T (-f_k \nabla^2 f_k)^{-1} \nabla f_k} \right\| \\
&\leq \|(\nabla^2 f_k)^{-1}\| \left(\|\nabla^2 f_k (x_k - x^*) - \nabla f_k\| + 2|\theta| \|\nabla f_k\| + |\theta| \|\nabla f_k\| \right) \\
&= \|(\nabla^2 f_k)^{-1}\| \left(\|\nabla^2 f_k (x_k - x^*) - (\nabla f_k - \nabla f^*)\| + 2|\theta| \|\nabla f_k\| + |\theta| \|\nabla f_k\| \right). \qquad (25)
\end{aligned}$$

Since

$$\nabla f_k - \nabla f^* = \int_0^1 \nabla^2 f(x_k + t(x^* - x_k))(x_k - x^*) dt, \qquad (26)$$

then

$$\begin{aligned}
&\|\nabla^2 f(x_k)(x_k - x^*) - (\nabla f_k - \nabla f^*)\| \\
&= \left\| \int_0^1 (\nabla^2 f(x_k) - \nabla^2 f(x_k + t(x^* - x_k)))(x_k - x^*) dt \right\| \\
&\leq \int_0^1 \|\nabla^2 f(x_k) - \nabla^2 f(x_k + t(x^* - x_k))\| \|(x_k - x^*)\| dt \\
&\leq \frac{1}{2} L \|x_k - x^*\|^2. \qquad (27)
\end{aligned}$$

Now from Lemma 3.4-**I**, Assumption 3.1 **(A2)** and (27), we obtain

$$\|x_{k+1} - x^*\| \leq \|(\nabla^2 f_k)^{-1}\| (\frac{1}{2} L \|x_k - x^*\|^2 + 2|\theta| L \|x_k - x^*\| + |\theta| L \|x_k - x^*\|. \qquad (28)$$

Since $\nabla^2 f(x^*)$ is nonsingular and $\nabla^2 f(x_k) \to \nabla^2 f(x^*)$, this implies that $\|(\nabla^2 f(x_k))^{-1}\| \leq 2\|(\nabla^2 f(x^*))^{-1}\|$ for all sufficiently large k. Therefore from Assumption 3.1 **(A4)**

$$\|x_{k+1} - x^*\| \leq \gamma (\|x_k - x^*\|^2 + 6|\theta| \|x_k - x^*\|). \qquad (29)$$

Now suppose that there exists an integer k such that (i)$\|x_k - x^*\| \leq 1$ or (ii) $\|x_k - x^*\| > 1$. Then we consider the following cases:
Case (i): This case implies that

$$\|x_{k+1} - x^*\| \leq \gamma \, (1 + 6|\theta|) \, \|x_k - x^*\|, \qquad (30)$$

and with the help of (30) we have

$$\|x_{k+1} - x^*\| \leq \gamma\,(1 + 6|\theta|)\,\|x_k - x^*\| \leq \cdots \leq (\gamma\,(1 + 6|\theta|))^{k+1}\|x_0 - x^*\|. \quad (31)$$

If the starting point is sufficiently near x^*, then, by using Assumption 3.1 **(A5)** the sequence of $\{x_k\}$ converges to x^*.
Case (ii): This case implies that

$$\|x_{k+1} - x^*\| \leq \gamma\,(1 + 6|\theta|)\,\|x_k - x^*\|^2, \quad (32)$$

and therefore

$$\|x_{k+1} - x^*\| \leq \gamma\,(1 + 6|\theta|)\,\|x_k - x^*\|^2 \leq \cdots \leq (\gamma\,(1 + 6|\theta|))^{2^{k+1}-1}\|x_0 - x^*\|^{2^{k+1}}, \quad (33)$$

and similar to previous case by using Assumption 3.1 **(A5)** the sequence of $\{x_k\}$ converges to x^*. □

Remark 1 For $\theta = 0$, the proposed method reduces to the standard Newton method.

4 Numerical results

In this section, we report some results on the following numerical experiments for the proposed algorithm. Also, the effectiveness of the proposed method with classical Newton method and Modified Regularized Newton method (MRN) proposed by Zhou and Chen in [8] are compared. We suppose $\tau = 10^{-6}$ and stopping condition is $\|g(x_k)\| \leq 10^{-5}$. Moreover, we suppose $p_0 = 0.0001$, $p_1 = 0.25$, $p_2 = 0.75$, $\mu_1 = 10^{-2}$ and $m = 10^{-8}$ in MRN method. N_f represents the number of the objective function evaluations and N_g represents the number of gradient evaluations. The codes have been written in Matlab 12.0 and runs are made on 2.3v GHz PC with 8 GB memory. The following test functions with standard starting points [26,27] and [28] are used for the comparison. The obtained numerical results are summarized in Tables 1 and 2. As shown in these tables, the proposed method (PM) is preferable to classical Newton method (NM) and MRN method. Here, we used the performance profiles proposed by Dolan and More [29] to compare and evaluate each implementation on the test functions. Figures 1, 2, 3, 4, 5 and 6 give the performance profiles of the proposed method with $\theta = 0.25, 0.5$ and 0.75, MRN method and the classical Newton method relative to the number of objective function evaluations (N_f) and the number of gradient evaluations (N_g). Also, "Dim" shows the dimension of the problem. Figures 1, 2 and 3 show the performance of considered algorithms in terms of number of function evaluations. At a glance to these figures, we observe that in the cases $\theta = 0.25$ and $\theta = 0.5$ the performance of the proposed algorithm is better than others. Also according to Figs. 4, 5 and 6, the proposed method with $\theta = 0.25$ performe better than the other methods.

Table 1 Numerical results

Problem/Dim	PM $\theta = 0.25$	PM $\theta = 0.5$	PM $\theta = 0.75$	MRN [8]	NM
	N_f	N_f	N_f	N_f	N_f
	N_g	N_g	N_g	N_g	N_g
	f^*	f^*	f^*	f^*	f^*
QUARTC/500	6	4	3	11	15
	7	5	4	23	16
	5.71e−5	2.78e−5	2.78e−5	7.98e−9	1.36e−8
NONSCOMP/500	6	5	6	7	9
	7	6	7	15	10
	7.89e−11	3.50e−11	1.55e−11	7.89e−30	2.12e−28
LIARWHD/500	8	7	9	9	12
	9	8	10	19	13
	4.86e−12	1.55e−8	1.04e−8	3.67e−24	1.28e−25
Broyden Tridiagonal/700	7	10	11	5	7
	8	11	12	11	8
	4.40e−13	8.73e−13	2.43e−12	3.52e−29	1.31e−28
Perturbed Tridiagonal Quadratic/400	1	1	1	3	2
	2	2	2	7	3
	1.31e−24	5.23e−24	1.18e−23	2.02e−53	4.57e−60
Extended Powell/100	18	20	35	13	17
	19	21	36	27	18
	1.82e−10	1.04e−9	9.71e−10	7.03e−7	4.27e−9
TRIDIA/500	1	1	1	4	2
	2	2	2	9	3
	1.75e−20	6.98e−20	1.57e−19	5.18e−32	2.06e−38
NONDQUAR/300	13	18	37	153	17
	14	19	38	307	18
	5.63e−9	5.45e−9	3.10e−8	2.39e−4	3.16e−10
ARWHEAD/500	7	1	13	5	Fail
	8	2	14	11	-
	4.55e−13	−2.27e−13	7.28e−12	−1.59e−12	-
Diagonal 4/300	1	1	1	4	2
	2	2	2	9	3
	1.38e−23	5.53e−23	1.24e−22	0	0
Extended Trigonometric/10	22	Fail	Fail	3	4
	23	-	-	7	5
	9.73e+2	-	-	9.73e+2	9.73e+2

Table 2 Numerical results

Problem/Dim	PM $\theta = 0.25$	PM $\theta = 0.5$	PM $\theta = 0.75$	MRN [8]	NM
	N_f	N_f	N_f	N_f	N_f
	N_g	N_g	N_g	N_g	N_g
	f^*	f^*	f^*	f^*	f^*
Extended Tridiagonal 1/200	13	16	22	11	14
	14	17	23	23	15
	1.62e−8	1.57e−8	3.88e−8	1.60e−9	1.38e−8
Generalized Tridiagonal 2/300	12	15	16	6	8
	13	16	17	13	9
	2.04e−13	2.81e−12	8.66e−11	4.03e−29	1.23e−24
POWER/500	1	1	1	4	2
	2	2	2	9	3
	1.71e−20	6.85e−20	1.55e−19	1.43e−44	0
NONDIA/100	1	1	1	4	70
	2	2	2	9	3
	8.52e−27	2.79e−26	8.93e−27	9.78e−23	1.97e−31
Tridiagonal Double Borded/200	12	13	Fail	6	24
	13	14	-	13	13
	4.43e−13	6.52e−11	-	1.35e−25	1.85e−17
Diagonal Double Bounded Arrow Up/300	10	31	27	Fail	11
	11	32	28	-	11
	7.97e−14	2.27e−13	1.51e−12	-	1.52e−25
DENSCHNF/100	7	4	15	5	7
	8	5	16	11	8
	2.10e−14	2.23e−16	1.94e−13	0	0

4.1 Systems of nonlinear equations

In this part, we solve systems of nonlinear equations by using the proposed algorithm. Consider the following nonlinear system of equations

$$F(x) = 0, \tag{34}$$

where $F(x) = (f_1(x), f_2(x), \ldots, f_n(x))$ and $x \in \mathbb{R}^n$. This system can be extended as

$$\begin{aligned} &f_1(x_1, x_2, \ldots, x_n) = 0, \\ &f_2(x_1, x_2, \ldots, x_n) = 0, \\ &\vdots \\ &f_n(x_1, x_2, \ldots, x_n) = 0. \end{aligned} \tag{35}$$

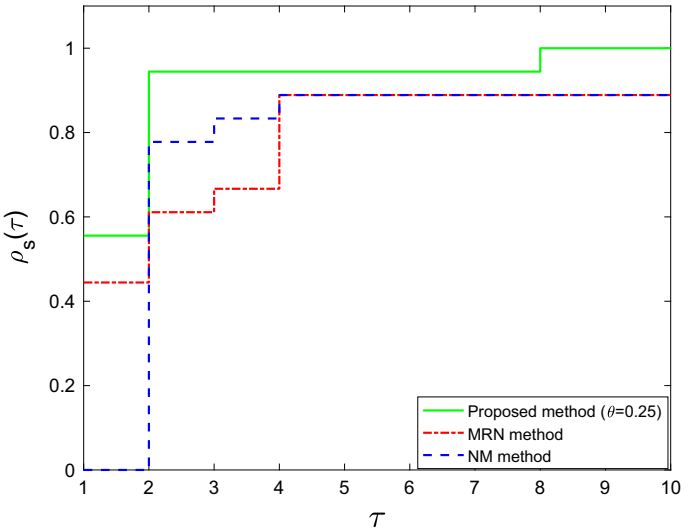

Fig. 1 Performance profile for the number of function evaluations with $\theta = 0.25$

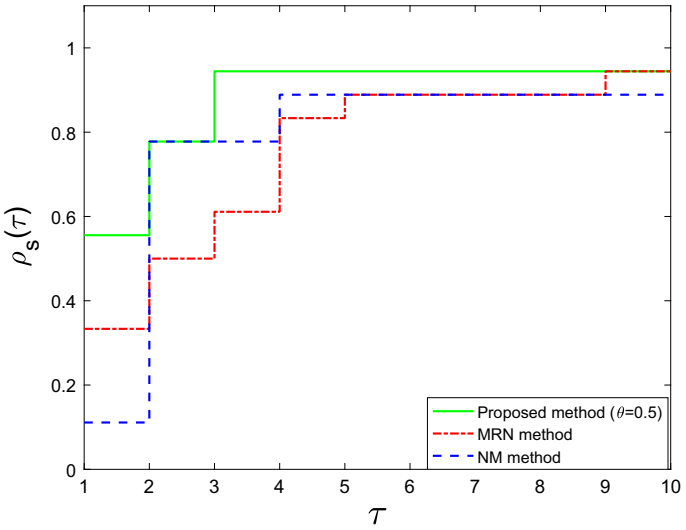

Fig. 2 Performance profile for the number of function evaluations with $\theta = 0.5$

For solving (34) by proposed algorithm, we suppose that $f(x) = \sum_{i=1}^{n} f_i^2(x)$. Here, we have worked out our proposed method on the following test problems. In all problems, the stopping criterion is given by $\|f(x_k)\| < 10^{-15}$.

Problem 1 Consider the system of two equations [30]:

$$F(x) = \begin{cases} x_1 + e^{x_2} - \cos x_2 = 0, \\ 3x_1 - \sin x_1 - x_2 = 0, \end{cases}$$

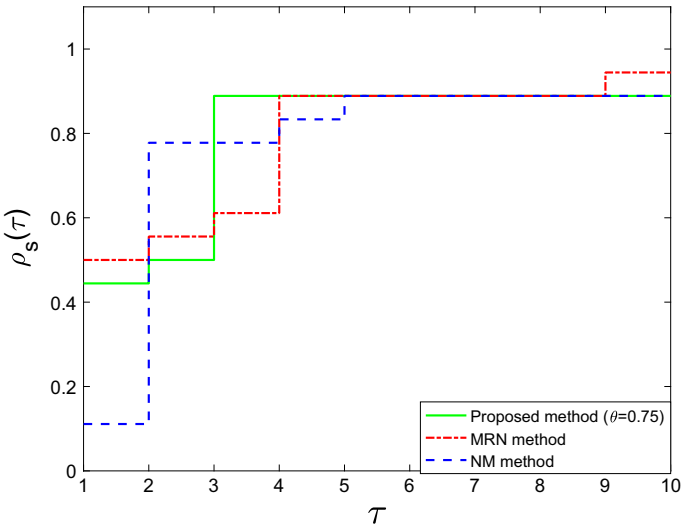

Fig. 3 Performance profile for the number of function evaluations with $\theta = 0.75$

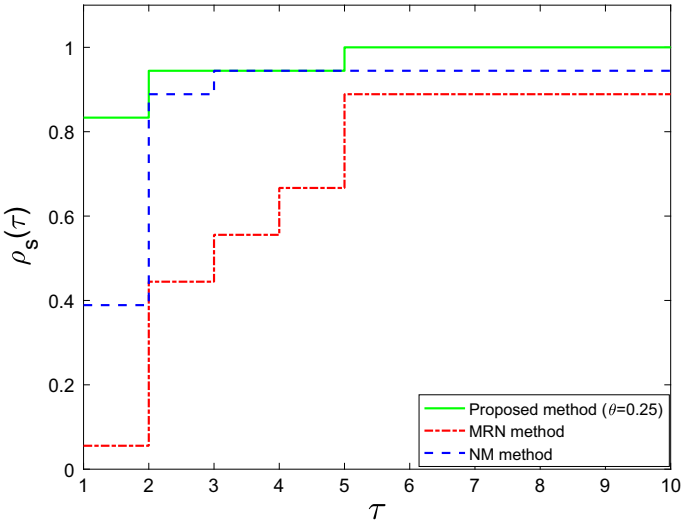

Fig. 4 Performance profile for the number of gradient evaluations with $\theta = 0.25$

with the initial value $x_0 = (1.5, 1.5)$ and exact solution $x^* = (0, 0)$.

Problem 2 Consider the system of three equations [30]:

$$F(x) = \begin{cases} 10x_1 + \sin(x_1 + x_2) - 1 = 0, \\ 8x_2 - \cos^2(x_3 - x_2) - 1 = 0, \\ 12x_3 + \sin x_3 - 1 = 0, \end{cases}$$

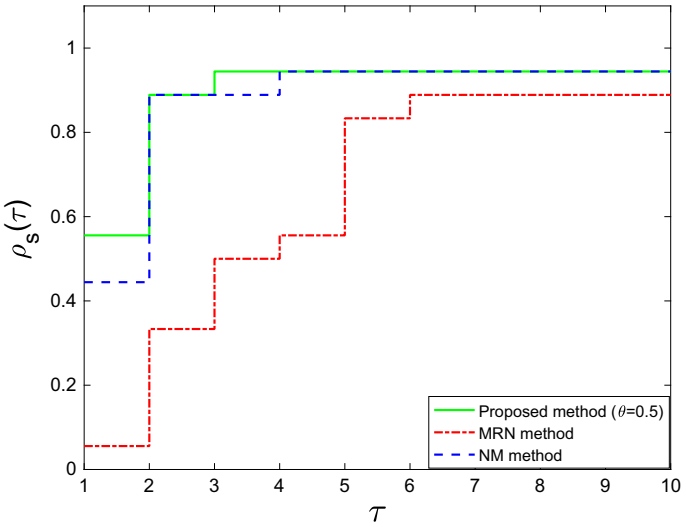

Fig. 5 Performance profile for the number of gradient evaluations with $\theta = 0.5$

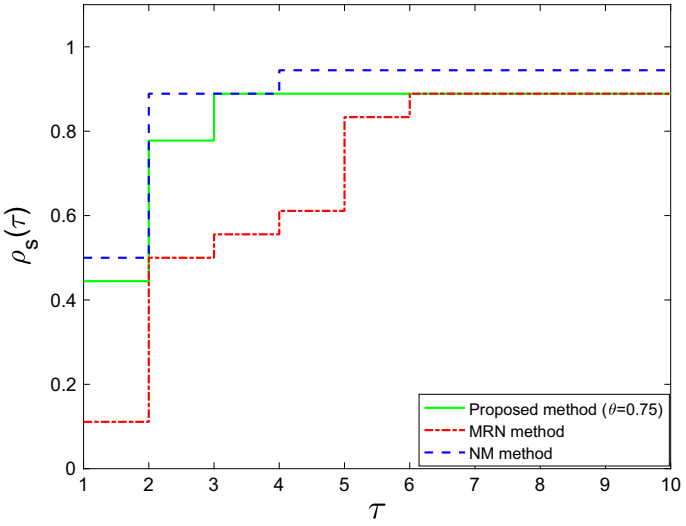

Fig. 6 Performance profile for the number of gradient evaluations with $\theta = 0.75$

with the initial value $x_0 = (-1, 1, -1)$ and exact solution

$$x^* = (0.0689783491726666, 0.2464424186091830, 0.0769289119875370).$$

Problem 3 $n = 16$, $1 \leq i \leq n-1$ [31]:

$$F(x) = \begin{cases} x_i \sin(x_{i+1}) - 1 = 0, \\ x_n \sin(x_1) - 1 = 0, \end{cases}$$

Table 3 Numerical results for Problems 1–4

Problem	PM $\theta = 0.25$	PM $\theta = 0.5$	PM $\theta = 0.75$	MRN method [8]	NM
	N_h	N_h	N_h	N_h	N_h
	N_g	N_g	N_g	N_g	N_g
	CPU time(s)	CPU time(s)	CPU time(s)	CPU time(s)	CPU time(s)
	$\|f_k - f^*\|$	$\|f_k - f^*\|$	$\|f_k - f^*\|$	$\|f_k - f^*\|$	$\|f_k - f^*\|$
1	6	6	8	7	114
	5	5	7	13	113
	0.156	0.129	0.178	0.173	1.789
	8.26e−11	2.50e−9	7.72e−15	8.80e−18	6.01e−16
2	6	6	6	4	21
	5	5	5	9	20
	0.164	0.168	0.188	0.140	0.421
	7.72e−14	9.13e−16	3.93e−17	7.47e−17	4.99e−17
3	4	4	4	4	13
	3	3	3	7	12
	0.204	0.213	0.212	0.193	0.391
	1.61e−8	4.44e−11	0	8.88e−16	8.90e−16
4	4	4	4	3	5
	3	3	3	5	4
	46.102	42.072	46.534	33.938	0.862
	6.87e−12	9.42e−16	8.19e−16	4.29e−16	2.07e−4

with the initial value $x_0 = (-0.85, -0.85, \ldots, -0.85)$ and exact solution

$$x^* = (-1.114157140871930087, \ldots, -1.114157140871930087).$$

Problem 4 Next, consider a system of 25 equations [32]:

$$f_i(x) = x_i - \cos\left(2x_i - \sum_{j=1}^{25} x_j\right) = 0, \quad 1 \le i \le 25.$$

The initial guess is $x_0 = (0.2, 0.2, \ldots, 0.2)$ and the root correct up to 16 digits is given by $x^* = (0.2142756147552158, 0.2142756147552158, \ldots, 0.2142756147552158)$.

Table 3 shows the results about the considered nonlinear systems solved by the proposed algorithm, MRN and classical Newton method. From Table 3, we can see that the performance of proposed method on these problems is better than the other two algorithms.

5 Conclusions

In the present work, we proposed a new algorithm for solving convex unconstrained optimization problems. This proposed method is based on Traub's iterative algorithm [1] which is extended to n-variable. The local convergence of the proposed method is also provided. The best property of this method is that it does not need to calculate the step length and $\alpha_k = 1$ at each iteration. The numerical results and comparison with other algorithms confirmed the efficiency and robustness of our algorithm. In addition, by comparing the performance profile figures for three different $\theta = 0.25, 0.5, 0.75$, it is seen that the numerical results for $\theta = 0.25$ are better than two other θ values. This algorithm is also used to solve systems of nonlinear equations, which have better numerical results compared to MRN method and classical Newton method.

References

1. Traub, J.F.: Iterative Methods for the Solution of Equations. Prentice Hall, Englewood Cliffs (1964)
2. Nocedal, J., Wright, S.: Numerical Optimization, 2nd edn. Springer, New York (2006)
3. Kelley, C.T.: Iterative Methods for Linear and Nonlinear Equations. North Carolina State University, Chapel Hill (1995)
4. Dennis, J.E., Schnabel, R.B.: Numerical Methods for Unconstrained Optimization and Nonlinear Equation. Society for Industrial and Applied Mathematics, Prentice-Hall (1996)
5. Zhou, G., Toh, K.C.: Superlinear convergence of a Newton-type algorithm for monotone equations. J. Optim. Theory Appl. **125**, 205–221 (2005)
6. Fan, J.Y.: Convergence rate of the trust region method for nonlinear equations under local error bound condition. Comput. Optim. Appl. **34**, 215–227 (2006)
7. Li, D.H., Fukushima, M., Qi, L., Yamashita, N.: Regularized Newton methods for convex minimization problems with singular solutions. Comput. Optim. Appl. **28**, 131–147 (2004)
8. Zhou, W., Chen, X.: On the convergence of a modified regularized Newton method for convex optimization with singular solutions. Comput. Appl. Math. **239**, 179–188 (2013)
9. Nesterov, Y., Nemirovsky, A.: Interior Point Polynomial Algorithms in Convex Programming. SIAM, Philadelphia (1994)
10. Nesterov, Y., Polyak, B.T.: Cubic regularization of Newton method and its global performance. Math. Program. **108**, 177–205 (2006)
11. Polyak, R.A.: Regularized Newton method for unconstrained convex optimization. Math. Program. **120**, 125–145 (2009)
12. Dehghan Niri, T., Hosseini, M.M., Heydari, M.: An efficient improvement of the Newton method for solving nonconvex optimization problems. Comput. Methods Differ. Equ. **7**, 69–85 (2019)
13. Heydari, M., Dehghan Niri, T., Hosseini, M.M.: A new modified trust region algorithm for solving unconstrained optimization problems. J. Math. Ext. **12**, 115–135 (2018)
14. Dehghan Niri, T., Shahzadeh Fazeli, S.A., Hosseini, M.M.: Using a new regularized factorization method for unconstrained optimization problems. Int. J. Numer. Model. Electron. Netw. Devices Fields (2019). https://doi.org/10.1002/jnm.2580
15. Abu Arqub, O., Al-Smadi, M.: Atangana–Baleanu fractional approach to the solutions of Bagley–Torvik and Painlevé equations in Hilbert space. Chaos Solitons Fractals **117**, 161–167 (2018)
16. Abu Arqub, O., Maayah, B.: Numerical solutions of integrodifferential equations of Fredholm operator type in the sense of the Atangana–Baleanu fractional operator. Chaos Solitons Fractals **117**, 117–124 (2018)
17. Abu Arqub, O.: Solutions of time-fractional Tricomi and Keldysh equations of Dirichlet functions types in Hilbert space. Numer. Methods Partial Differ. Equ. **34**, 1759–1780 (2018)
18. Abu Arqub, O.: Numerical solutions for the Robin time-fractional partial differential equations of heat and fluid flows based on the reproducing kernel algorithm. Int. J. Numer. Methods Heat Fluid Flow **28**, 828–856 (2018)

19. Abu Arqub, O.: Fitted reproducing kernel Hilbert space method for the solutions of some certain classes of time-fractional partial differential equations subject to initial and Neumann boundary conditions. Comput. Math. Appl. **73**, 1243–1261 (2017)
20. Aslam Noor, M.: New classes of iterative methods for nonlinear equations, A new modified Halley method without second derivatives for nonlinear equation. Appl. Math. Comput. **191**, 128–131 (2007)
21. Aslam Noor, M., Inayat Noor, K., Mohyud-Din, S.T., Shabbir, A.: An iterative method with cubic convergence for nonlinear equations. Appl. Math. Comput. **183**, 1249–1255 (2006)
22. Aslam Noor, M., Asghar Khan, W., Hussain, A.: A new modified Halley method without second derivatives for nonlinear equation. Appl. Math. Comput. **189**, 1268–1273 (2007)
23. Atkinson, K.E.: An Introduction to Numerical Analysis, 2nd edn. Wiley, Singapore (1988)
24. Duncan, W.J.: Some devices for the solution of large sets of simultaneous linear equations (with an appendix on the reciprocation ofp artitioned matrices). Lond. Edinb. Dublin Philos. Mag. J. Sci. **35**, 660–670 (1944)
25. Woodbury, M.A.: Inverting Modified Matrices, Memorandum Report 42. Statistical Research Group, Princeton, NJ (1950)
26. Andrei, N.: Test Functions for Unconstrained Optimization. Academy of Romanian Scientists, Bucharest (2004)
27. Andrei, N.: An unconstrained optimization test functions collection. Adv. Model. Optim. **10**, 147–161 (2008)
28. More, J.J., Grabow, B.S., Hillstrom, K.E.: testing unconstrained optimization software. ACM Trans. Math. Softw. **7**, 17–41 (1981)
29. Dolan, E.D., More, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201–213 (2002)
30. Sharma, J.R., Gupta, P.: An efficient fifth order method for solving systems of nonlinear equations. Comput. Math. Appl. **67**, 591–601 (2014)
31. Xiao, X., Yin, H.: A new class of methods with higher order of convergence for solving systems of nonlinear equations. Appl. Math. Comput. **264**, 300–309 (2015)
32. Narang, M., Bhatia, S., Kanwar, V.: New two-parameter Chebyshev–Halley-like family of fourth and sixth-order methods for systems of nonlinear equations. Appl. Math. Comput. **275**, 394–403 (2016)