Two globally convergent nonmonotone trust-region methods for unconstrained optimization

Masoud Ahookhosh · Susan Ghaderi

Received: 15 January 2015 / Published online: 26 March 2015 © Korean Society for Computational and Applied Mathematics 2015

Abstract This paper addresses some trust-region methods equipped with nonmonotone strategies for solving nonlinear unconstrained optimization problems. More specifically, the importance of using nonmonotone techniques in nonlinear optimization is motivated, then two new nonmonotone terms are proposed, and their combinations into the traditional trust-region framework are studied. The global convergence to first- and second-order stationary points and local superlinear and quadratic convergence rates for both algorithms are established. Numerical experiments on the CUTEst test collection of unconstrained problems and some highly nonlinear test functions are reported, where a comparison among state-of-the-art nonmonotone trust-region methods shows the efficiency of the proposed nonmonotone schemes.

Keywords Unconstrained optimization · Black-box oracle · Trust-region method · Nonmonotone strategy · Global convergence

Mathematics Subject Classification 90C30 · 65K05
1 Introduction

In this paper we consider the unconstrained minimization problem

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad x \in \mathbb{R}^n,
\end{align*}
\]

where \( f : \mathbb{R}^n \to \mathbb{R} \) is a real-valued nonlinear function, which is bounded and continuously-differentiable. We suppose that first- or second-order black-box oracle of \( f \) is available.

1.1 Motivation and history

Trust-region methods, also called restricted step methods [22], are a class of iterative schemes developed to solve convex or nonconvex optimization problems, see, for example, [14]. They also developed for nonsmooth problems, see [14, 17, 24, 46]. Trust-region methods have strong convergence properties, are reliable and robust in computation, and can handle ill-conditioned problems, cf. [35, 36]. Let \( x_k \) be the current iteration. In trust-region framework the objective \( f \) is approximated by a simple model in a specific region around \( x_k \) such that it is an acceptable approximation of the original objective, which is called trust-region (region of trust). Afterward, the model is minimized subject to the trust-region constraint to find a new trial point \( d_k \). Hence the simple model means that it can be minimized much easier than the original objective function. If the founded model is an adequate approximation of the objective function within the trust-region, then the point \( x_{k+1} = x_k + d_k \) is accepted by the trust-region method and the region can be expanded for the next iteration; conversely, if the approximation is poor, the region is contracted and the model is minimized within the contracted region. This scheme will be continued until finding an acceptable trial step \( d_k \) guaranteeing an acceptable agreement between the model and the objective function.

Several quadratic and non-quadratic models have been proposed to approximate the objective function in optimization, see [15, 23, 37, 40], however, the conic and quadratic models are becoming more popular, see [18, 37, 38]. If the approximated model is quadratic, i.e.,

\[
q_k(d) := f_k + g_k^T d + \frac{1}{2} d^T B_k d,
\]

where \( f_k = f(x_k), g_k = \nabla f(x_k), \) and \( B_k \approx H(x_k) = \nabla^2 f(x_k), \) the trust-region method can be considered as a globally convergent generalization on classical Newton’s method. Then the trust-region subproblem is defined by

\[
\begin{align*}
\text{minimize} & \quad q_k(d), \\
\text{subject to} & \quad \|d\| \leq \delta_k.
\end{align*}
\]

Hence the trust-region is commonly a norm ball \( C \) defined by

\[
C := \{ d \in \mathbb{R}^n \mid \|d\| \leq \delta_k \},
\]
where $\delta_k > 0$ is a real number called trust-region radius, and $\| \cdot \|$ is any norm in $\mathbb{R}^n$, cf. [30]. Since $C$ is compact and the model is continuous, the trust-region subproblem attains its minimizer on the set $C$. The most computational cost of trust-region methods relates to minimizing the model over the trust-region $C$. Once the step $d$ is computed, the quality of the model in the trust-region is evaluated by a ratio of the actual reduction of objective, $f_k - f(x_k + d)$, to the predicted reduction of model, $q_k(0) - q_k(d)$, i.e.,

$$r_k = \frac{f_k - f(x_k + d)}{q_k(0) - q_k(d)}. \quad (4)$$

For a prescribed positive constant $\mu_1 \in (0, 1]$, if $r_k \geq \mu_1$, the model provides a reasonable approximation, the step is accepted, i.e., $x_{k+1} = x_k + d_k$, and the trust-region $C$ can be expanded for the next step. Otherwise, the trust-region $C$ should be contracted by decreasing the radius $\delta_k$ and the subproblem (3) is solved in the reduced region. This scheme is continued until that the step $d$ accepted by trust-region test $r_k \geq \mu_1$. Our discussion can be summarized in the following algorithm:

**Algorithm 1: TTR (traditional trust-region algorithm)**

**Input:** $x_0 \in \mathbb{R}^n$, $B_0 \in \mathbb{R}^{n \times n}$, $k_{\text{max}}$; $0 < \mu_1 \leq \mu_2 \leq 1$, $0 < \rho_1 < 1 \leq \rho_2$, $\epsilon > 0$;

**Output:** $x_b$; $f_b$;

1. begin
   2. $\delta_0 \leftarrow \|g_0\| \; ; \; k \leftarrow 0$;
   3. while $\|g_k\| \geq \epsilon$ & $k \leq k_{\text{max}}$ do
      4. solve the subproblem (3) to specify $d_k$;
      5. $\hat{x}_k \leftarrow x_k + d_k$; compute $f(\hat{x}_k)$;
      6. determine $r_k$ using (4);
      7. while $r_k < \mu_1$ do
         8. $\delta_k \leftarrow \rho_1 \delta_k$;
         9. solve the subproblem (3) to specify $d_k$;
        10. $\hat{x}_k \leftarrow x_k + d_k$; compute $f(\hat{x}_k)$
        11. determine $r_k$ using (4);
    12. end
   13. $x_{k+1} \leftarrow \hat{x}_k$;
   14. if $r_k \geq \mu_2$ then
      15. $\delta_{k+1} \leftarrow \rho_2 \delta_k$;
   16. end
   17. update $B_{k+1}$; $k \leftarrow k + 1$;
   18. end
   19. $x_b \leftarrow x_k$; $f_b \leftarrow f_k$;
   20. end

In Algorithm 1, it follows from $r_k \geq \mu_1$ and $q_k(0) - q_k(d_k) > 0$ that

$$f_k - f_{k+1} \geq \mu_1(q_k(0) - q_k(d_k)) > 0,$$

© Springer
implying $f_{k+1} \leq f_k$. This means that the sequence of function values $\{f_k\}$ is monotonically decreasing, i.e., the traditional trust-region method is also called the **monotone trust-region** method. This feature seems natural for minimization schemes, however, it slows down the convergence of TTR to a minimizer if the objective involves a curved narrow valley, see [1,28]. To observe the effect of nonmonotonicity on TTR, we study the next example.

**Example 1** Consider the two-dimensional Nesterov–Chebyshev–Rosenbrock function, cf. [29],

$$f(x_1, x_2) = \frac{1}{4}(x_1 - 1)^2 + (x_2 - 2x_1^2 + 1)^2,$$

where we solve the problem (1) by Newton’s method and TTR with the initial point $x_0 = (-0.61, -1)$. It is clear that $(1,1)$ is the optimizer. The implementation indicates that Newton’s method needs 7 iterations and 8 function evaluations, while monotone trust-region method needs 22 iterations and 24 function evaluations. We depict the contour plot of the objective and iteration points as well as a diagram for function values versus iteration attained by these two algorithms in Fig. 1. Figure 1a shows that the iterations of TTR follow the bottom of the valley in contrast to those for Newton’s method that can go up and down to reach the $\varepsilon$-solution with the accuracy parameter $\varepsilon = 10^{-5}$. We see that Newton’s method attains larger step compared with those of TTR. Figure 1b illustrates function values versus iterations for both algorithms showing that the related function values of TTR decreases monotonically, while it is fluctuated nonmonotonically for Newton’s method.

In general the monotonicity may result to the slow iterative schemes for highly nonlinear or badly-scaled problems. To avoid this algorithmic limitation, the idea of nonmonotone strategies has been proposed traced back to the watch-dog technique to overcome the Maratos effect for constrained optimization [13]. To improve the
performance of Armijo’s line search, Grippo et al. in 1986 [28] proposed the modified Armijo’s rule
\[ f(x_k + \alpha_k d_k) \leq f_i(k) + \sigma \alpha_k g_k^T d_k, \quad k = 0, 1, 2, \ldots, \]
with the step-size \( \alpha_k > 0, \sigma \in (0, 1/2) \), and
\[ f_i(k) = \max_{0 \leq j \leq m(k)} \{ f_{k-j} \}, \quad (5) \]
where \( m(0) = 0, m(k) \leq \min\{m(k-1) + 1, N\} \) for nonnegative integer \( N \). It was shown that the associated scheme is globally convergent, and numerical results reported in Grippo et al. [28] and Toint [41] showed the effectiveness of the proposed idea. Motivated by these results, the nonmonotone strategies has received much attention during past few decades. For example, in 2004, Zhang and Hager in [47] proposed the nonmonotone term
\[ C_k = \begin{cases} f_0 & \text{if } k = 0, \\ (\eta_{k-1} Q_{k-1} C_{k-1} + f(x_k))/Q_k & \text{if } k \geq 1, \end{cases} \]
\[ Q_k = \begin{cases} 1 & \text{if } k = 0, \\ \eta_{k-1} Q_{k-1} + 1 & \text{if } k \geq 1, \end{cases} \quad (6) \]
where \( 0 \leq \eta_{\min} \leq \eta_{k-1} \leq \eta_{\max} \leq 1 \). Recently, Mo et al. in [31] and Ahookhosh et al. in [3] studied the nonmonotone term
\[ D_k = \begin{cases} f_k & \text{if } k = 1, \\ \eta_k D_{k-1} + (1 - \eta_k) f_k & \text{if } k \geq 2, \end{cases} \quad (7) \]
where \( \eta_k \in [\eta_{\min}, \eta_{\max}], \eta_{\min} \in [0, 1], \eta_{\max} \in [\eta_{\min}, 1] \). More recently, Amini et al. in [8] proposed the nonmonotone term
\[ R_k = \eta_k f_i(k) + (1 - \eta_k) f_k, \quad (8) \]
where \( 0 \leq \eta_{\min} \leq \eta_{\max} \leq 1 \) and \( \eta_k \in [\eta_{\min}, \eta_{\max}] \). In all cases it was proved that the schemes are globally convergent and enjoy the better performance compared with monotone ones.

At the same importance of using nonmonotone strategies for inexact line search techniques, the combination of trust-region methods with nonmonotone strategies is interesting. Historically, the first nonmonotone trust-region method was proposed in 1993 by Deng et al. in [16] for unconstrained optimization. Under some classical assumptions, the global convergence and the local superlinear convergence rate were established. Nonmonotone trust-region methods were also studied by several authors such as Toint [42], Xiao and Zho [44], Xiao and Chu [45], Zhou and Xiao [48], Ahookhosh and Amini [2], Amini and Ahookhosh [7], and Mo et al. [31]. Recently, Ahookhosh and Amini in [1] and Ahookhosh et al. in [4] proposed two nonmonotone trust-region methods using the nonmonotone term (8). Theoretical results were reported, and numerical results showed the efficiency of the proposed nonmonotone methods.
1.2 Content

In this paper we propose a trust-region method equipped with two novel nonmonotone terms. More precisely, we first establish two nonmonotone terms and then combine them with Algorithm 1 to construct two nonmonotone trust-region algorithms. If \( k \geq N \), the new nonmonotone terms are defined by a convex combination of the last \( N \) successful function values, and if \( k < N \), either a convex combination of \( k \) successful function values or \( f_{i(k)} \) is used. The global convergence to first- and second-order stationary points is established on some classical assumptions. Moreover, local superlinear and quadratic convergence rates for the proposed methods are studied. Numerical results regarding experiments on some highly nonlinear problems and on 112 unconstrained test problems from the CUTEst test collection [25] are reported indicating the efficiency of the proposed nonmonotone terms.

The remainder of paper is organized as follow. In Sect. 2 we propose new nonmonotone terms and their combination with the trust-region framework. The global convergence of the proposed methods are given in Sect. 3. Numerical results are reported in Sect. 4. Finally, some conclusions are given in Sect. 5.

2 Novel nonmonotone terms and algorithm

In this section we first present two novel nonmonotone terms and then combine them into trust-region framework to introduce two nonmonotone trust-region algorithms for solving the unconstrained optimization problem (1).

We first assume that \( k \) denotes the current iteration and \( N \in \mathbb{N} \) is a constant. The main idea is to construct a nonmonotone term determined by a convex combination of the last \( k \) successful function values if \( k < N \) and by a convex combination of the last \( N \) successful function values if \( k \geq N \). In the other words, we construct new terms using function values collected in the set

\[
\mathcal{F}_k := \begin{cases} 
\{f_0, f_1, \ldots, f_k\} & \text{if } k < N, \\
\{f_{k-N+1}, f_{k-N+2}, \ldots, f_k\} & \text{if } k \geq N,
\end{cases}
\]  

(9)

which should be updated in each iteration. To this end, motivated by the term (7), we construct \( T_k \) using the subsequent procedure

\[
\begin{align*}
\bar{T}_0 &= f_0, \\
\bar{T}_1 &= (1 - \eta_0)f_1 + \eta_0 f_0, \\
\bar{T}_2 &= (1 - \eta_1)f_2 + \eta_1(1 - \eta_0)f_1 + \eta_1\eta_0 f_0, \\
& \quad \vdots \\
\bar{T}_{N-1} &= (1 - \eta_{N-2})f_{N-1} + \eta_{N-2}(1 - \eta_{N-3})f_{N-2} + \cdots + \eta_0 f_0, \\
\bar{T}_N &= (1 - \eta_{N-1})f_N + \eta_{N-1}(1 - \eta_{N-2})f_{N-1} + \cdots + \eta_0 f_0.
\end{align*}
\]
where \( \eta_i \in \{0, 1\} \), for \( i = 1, 2, \ldots, N \), are some weight parameters. Hence the new term is generated by

\[
\overline{T}_k := \begin{cases} 
(1 - \eta_{k-1}) f_k + \eta_{k-1} \overline{T}_{k-1} & \text{if } k < N, \\
(1 - \eta_{k-1}) f_k + \eta_{k-1}(1 - \eta_{k-2}) & \\
f_{k-1} + \cdots + \eta_{k-1} \cdots \eta_{k-N} f_{k-N} & \text{if } k \geq N,
\end{cases}
\]

where \( \overline{T}_0 = f_0 \) and \( \eta_i \in \{0, 1\} \) for \( i = 1, 2, \ldots, k \). To show that \( \overline{T}_k \) is a convex combination of the collected function values \( \mathcal{F}_k \), it is enough to show that the summation of multipliers are equal to unity. For \( k \geq N \), the definition for \( \overline{T}_k \) implies

\[
(1 - \eta_{k-1}) + \eta_{k-1}(1 - \eta_{k-2}) + \cdots + \eta_{k-1} \cdots \eta_{k-N-1}(1 - \eta_{k-N}) + \eta_{k-1} \cdots \eta_{k-N} = 1
\]

For \( k < N \), a similar summation of the last \( k \) multipliers is equal to one. Therefore, the generated term \( \overline{T}_k \) is a convex combination of the elements of \( \mathcal{F}_k \).

The procedure of defining \( \overline{T}_k \) clearly implies that the set \( \mathcal{F}_k \) should be updated and saved in each iteration. Moreover, \( N(N + 1)/2 \) multiplications is required to compute \( \overline{T}_k \). To avoid saving \( \mathcal{F}_k \) and decrease the number of multiplications, we derive a recursive formula for (10). From the definition of \( \overline{T}_k \), for \( k \geq N \), it follows that

\[
\overline{T}_k - \eta_{k-1} \overline{T}_{k-1} = (1 - \eta_{k-1}) f_k + \eta_{k-1}(1 - \eta_{k-2}) f_{k-1} + \cdots + \eta_{k-1} \cdots \eta_{k-N} f_{k-N} - \eta_{k-1}(1 - \eta_{k-2}) f_{k-1} - \cdots - \eta_{k-1} \cdots (1 - \eta_{k-N-1}) f_{k-N} - \eta_{k-1} \eta_{k-2} \cdots \eta_{k-N-1} f_{k-N-1} = (1 - \eta_{k-1}) f_k + \eta_{k-1} \eta_{k-2} \cdots \eta_{k-N-1} (f_{k-N} - f_{k-N-1}) = (1 - \eta_{k-1}) f_k + \xi_k (f_{k-N} - f_{k-N-1})
\]

where \( \xi_k := \eta_{k-1} \eta_{k-2} \cdots \eta_{k-N-1} \). For \( k \geq N \), this equation leads to

\[
\overline{T}_k = (1 - \eta_{k-1}) f_k + \eta_{k-1} \overline{T}_{k-1} + \xi_k (f_{k-N} - f_{k-N-1}),
\]

which requires to save only \( f_{k-N} \) and \( f_{k-N-1} \) and only needs three multiplications. Moreover, the definition of \( \xi_k \) implies

\[
\xi_k = \eta_{k-1} \eta_{k-2} \cdots \eta_{k-N-1} = \frac{\eta_{k-1}}{\eta_{k-N-2}} \frac{\eta_{k-2}}{\eta_{k-N-3}} \cdots \eta_{k-N-3} = \frac{\eta_{k-1}}{\eta_{k-N-2}} \xi_{k-1}.
\]

If \( \xi_k \) is recursively updated by (13), (10), and (12), a new nonmonotone term is defined by
\[ T_k := \begin{cases} f_k + \eta_{k-1} (\bar{T}_k - f_k) & \text{if } k < N, \\ \max \{ \bar{T}_k, f_k \} & \text{if } k \geq N, \end{cases} \tag{14} \]

where the max term is added to guarantee \( T_k \geq f_k \).

As discussed in Sect. 1, nonmonotone schemes perform better when they use stronger nonmonotone terms far away from the optimizer and weaker one close to it. This motivate us to consider a new version of the derived nonmonotone term by using \( f_{l(k)} \) in cases that \( k < N \). More precisely, the second nonmonotone term is defined by

\[ T_k = \begin{cases} f_{l(k)} & \text{if } k < N, \\ \max \{ T_k, f_k \} & \text{if } k \geq N, \end{cases} \tag{15} \]

where \( \xi_k \) is defined by (13). It is clear that the new term uses a stronger term \( f_{l(k)} \) defined by (5) for first \( k < N \) iterations and then employs the relaxed convex term proposed above.

Now, to employ the proposed nonmonotone terms in the trust-region framework, it is enough to replace the ratio \( r_k \) (4) by the nonmonotone ratio

\[ \hat{r}_k = \frac{T_k - f(x_k + d)}{q_k(0) - q_k(d)}, \tag{16} \]

where \( T_k \) is defined by (14) or (15). Hence in trust-region framework we replace (4) by (16). Notice that if \( \hat{r}_k \geq \mu_1 \), then,

\[ T_k - f_{k+1} \geq \mu_1 (q_k(0) - q_k(d_k)) \geq 0. \]

This implies that \( f_{k+1} \) can be larger than \( f_k \), however, the elements of \( \{ f_k \} \) cannot arbitrarily increase, and the maximum increase is controlled by the nonmonotone term \( T_k \). Moreover, the definitions (14) and (15) imply that \( \hat{r}_k \geq r_k \) increasing the possibility of attaining larger steps for nonmonotone schemes compared with monotone ones.

The above-mentioned discussion leads to the following nonmonotone trust-region algorithm:
Algorithm 2: NMTR (nonmonotone traditional trust-region algorithm)

**Input**: $x_0 \in \mathbb{R}^n$, $B_0 \in \mathbb{R}^{n \times n}$, $k_{max}$; $0 < \mu_1 \leq \mu_2 \geq 1$, $0 < \rho_1 < 1 \leq \rho_2$, $\varepsilon > 0$;
**Output**: $x_b$; $f_b$;

1 begin
2 \[ \delta_0 \leftarrow \|g_0\|; \ k \leftarrow 0; \]
3 while \( \|g_k\| \geq \varepsilon \) \& \( k \leq k_{max} \) do
4 \[ \hat{x}_k \leftarrow x_k + d_k; \ \text{compute} \ f(\hat{x}_k); \]
5 determine $\hat{r}_k$ using (16);
6 if $\hat{r}_k < \mu_1$ while $\hat{r}_k < \mu_1$ do
7 \[ \delta_k \leftarrow \rho_1 \delta_k; \]
8 solve the subproblem (3) to specify $d_k$;
9 $\hat{x}_k \leftarrow x_k + d_k$; compute $f(\hat{x}_k)$;
10 determine $\hat{r}_k$ using (16);
11 end
12 \[ x_{k+1} \leftarrow \hat{x}_k; \]
13 if $\hat{r}_k \geq \mu_2$ then
14 \[ \delta_{k+1} \leftarrow \rho_2 \delta_k; \]
15 end
16 update $B_{k+1}$; \( k \leftarrow k + 1; \)
17 update $T_{k+1}$;
18 update $\eta_{k+1}$;
19 end
20 $x_b \leftarrow x_k$; $f_b \leftarrow f_k$;
21 end

In Algorithm 2, if $\hat{r}_k \geq \mu_1$ (Line 7), it is called a **successful** iteration and if $\hat{r}_k \geq \mu_2$ (Line 14), it is called a **very successful** iteration. In addition, in the algorithm, the loop started from Line 3 to Line 20 is called the **outer cycle**, and the loop started from Line 7 to Line 12 is called the **inner cycle**.

### 3 Convergence analysis

This section concerns with the global convergence to first- and second-order stationary points of the sequence $\{x_k\}$ generated by Algorithm 2. More precisely, we intend to prove that all limit point $x^*$ of the sequence $\{x_k\}$ satisfy the condition $g(x^*) = 0$, and there exists a point $x^*$ satisfying $g(x^*) = 0$ where $H(x^*)$ is positive semidefinite. Furthermore, we show that Algorithm 2 is well-defined, which means that the inner cycle of the algorithm will be left after a finite number internal iterations, and then prove its global convergence. Moreover, local superlinear and quadratic convergence rates are investigated under some classical assumptions.

To prove the global convergence of the sequence $\{x_k\}$ generated by Algorithm 2, we require the following assumptions:

**(H1)** The objective function $f$ is continuously differentiable and has a lower bound on the upper level set $L(x_0) = \{x \in \mathbb{R}^n \mid f(x) \leq f(x_0)\}$.
(H2) The sequence \( \{ B_k \} \) is uniformly bounded, i.e., there exists a constant \( M > 0 \) such that
\[
\| B_k \| \leq M,
\]
for all \( k \in \mathbb{N} \).

(H3) There exists a constant \( c > 0 \) such that the trial step \( d_k \) satisfies \( \| d_k \| \leq c \| g_k \| \), for all \( k \in \mathbb{N} \).

We also assume that the decrease on the model \( q_k \) is at least as much as a fraction of the decrease obtained by the Cauchy point guaranteeing that there exists a constant \( \beta \in (0, 1) \) such that
\[
q_k(0) - q_k(d_k) \geq \beta \| g_k \| \min \left\{ \delta_k, \frac{\| g_k \|}{\| B_k \|} \right\}, \tag{17}
\]
for all \( k \). This condition is called the sufficient reduction condition. Inequality (17) implies that \( d_k \neq 0 \) whenever \( g_k \neq 0 \). It is noticeable that there are several schemes that can solve the trust-region subproblem (3) such that (17) is valid, see, for example, [14, 33].

**Lemma 2** Suppose that the sequence \( \{ x_k \} \) is generated by Algorithm 2, then
\[
| f_k - f(x_k + d_k) - (q_k(0) - q_k(d_k)) | \leq O(\| d_k \|^2).
\]

*Proof* The proof can be found in [13]. \( \square \)

**Lemma 3** Suppose that the sequence \( \{ x_k \} \) is generated by Algorithm 2, then we get
\[
f_k \leq T_k \leq f_{i(k)}, \tag{18}
\]
for all \( k \in \mathbb{N} \cup \{0\} \).

*Proof* For \( k < N \), we consider two cases: (i) \( T_k \) is defined by (14); (ii) \( T_k \) is defined by (15). In Case (i) Lemma 2.1 in [3], \( f_i \leq f_{i(k)} \), for \( i = 0, 1, \ldots, k \), and the fact that summation of multipliers in \( T_k \) equal to one give the result. Case (ii) is evident from (15).

For \( k \geq N \), if \( T_k = f_k \), the result is evident. Otherwise, since
\[
(1 - \eta_{k-1}) + \eta_{k-1}(1 - \eta_{k-2}) + \cdots + \eta_{k-1} \cdots \eta_{k-N} (1 - \eta_{k-N}) + \eta_{k-1} \cdots \eta_{k-N} = 1, \tag{19}
\]
the fact that \( f_i \leq f_{i(k)} \), for \( i = k - N + 1, \ldots, k \), and (10) imply
\[
f_k \leq T_k = (1 - \eta_{k-1}) f_k + \eta_{k-1}(1 - \eta_{k-2}) f_{k-1} + \cdots + \eta_{k-1} \cdots \eta_{k-N} f_{k-N} \leq \left[ (1 - \eta_{k-1}) + \eta_{k-1}(1 - \eta_{k-2}) + \cdots + \eta_{k-1} \cdots \eta_{k-N} \right] f_{i(k)} = f_{i(k)},
\]
giving the result. \( \square \)
Lemma 4 Suppose that the sequence \( \{x_k\} \) is generated by Algorithm 2, then the sequence \( \{f_i(k)\} \) is decreasing.

Proof The condition (18) implies that \( T_k \leq f_i(k) \). If \( x_{k+1} \) is accepted by Algorithm 2, then

\[
\frac{f_i(k) - f(x_k + d_k)}{q_k(0) - q_k(d_k)} \geq \frac{T_k - f(x_k + d_k)}{q_k(0) - q_k(d_k)} \geq \mu_1,
\]

leading to

\[
f_i(k) - f(x_k + d_k) \geq \mu_1 (q_k(0) - q_k(d_k)) \geq 0, \quad \text{for all } k \in \mathbb{N},
\]

implying

\[
f_i(k) \geq f_{k+1}, \quad \text{for all } k \in \mathbb{N}. \tag{20}
\]

Now, if \( k \geq N \), by using \( m(k+1) \leq m(k) + 1 \) and (20), we get

\[
f_i(k+1) = \max_{0 \leq j \leq m(k+1)} \{f_{k-j+1}\} \leq \max_{0 \leq j \leq m(k)+1} \{f_{k-j+1}\}
\]

\[
= \max \{f_i(k), f_{k+1}\} \leq f_i(k).
\]

For \( k < N \), it is obvious that \( m(k) = k \). Since, for any \( k, f_k \leq f_0 \), it is clear that \( f_i(k) = f_0 \). Therefore, in both cases, the sequence \( \{f_i(k)\} \) is decreasing. \( \square \)

Lemma 5 Suppose that (H1) holds and the sequence \( \{x_k\} \) is generated by Algorithm 2, then \( L(x_0) \) involves \( \{x_k\} \).

Proof The definition of \( T_k \) indicates that \( T_0 = f_0 \). By induction, we assume that \( x_i \in L(x_0) \), for all \( i = 1, 2, \ldots, k \), and then prove that \( x_{k+1} \in L(x_0) \). From (18), we get

\[
f_{k+1} \leq T_{k+1} \leq f_i(k+1) \leq f_i(k) \leq f_0,
\]

implying that \( L(x_0) \) involves the sequence \( \{x_k\} \). \( \square \)

Corollary 6 Suppose that (H1) holds and the sequence \( \{x_k\} \) is generated by Algorithm 2. Then the sequence \( \{f_i(k)\} \) is convergent.

Proof The assumption (H1) and Lemma 4 imply that there exists a constant \( \lambda \) such that

\[
\lambda \leq f_{k+n} \leq f_i(k+n) \leq \cdots \leq f_i(k+1) \leq f_i(k),
\]

for all \( n \in \mathbb{N} \). This implies that the sequence \( \{f_i(k)\} \) is convergent. \( \square \)

Lemma 7 Suppose that (H1)–(H3) hold and the sequence \( \{x_k\} \) is generated by Algorithm 2, then

\[
\lim_{k \to \infty} f(x_{l(k)}) = \lim_{k \to \infty} f_k. \tag{21}
\]

Proof The condition (18) and Lemma 7 of [1] imply that the result is valid. \( \square \)
**Corollary 8** Suppose (H1)–(H3) hold and the sequence \( \{x_k\} \) is generated by Algorithm 2, then we have
\[
\lim_{k \to \infty} T_k = \lim_{k \to \infty} f_k.
\]

**Proof** From (18) and Lemma 7, the result is obtained. \( \square \)

**Lemma 9** Suppose that (H1) and (H2) hold, and the sequence \( \{x_k\} \) is generated by Algorithm 2. Then if \( \|g_k\| \geq \varepsilon > 0 \), we have

(i) The inner cycle is left after a finite number of inner iterations and Algorithm 2 is well-defined;

(ii) For any \( k \), there exists a nonnegative integer \( p \) such that \( x_{k+p+1} \) is a very successful iteration.

**Proof** (i) Let \( t_k \) denotes the internal iteration counter in step \( k \), and \( d_{tk}^k \) and \( \delta_{tk}^k \) respectively show the solution of the subproblem \( (3) \) and the corresponding trust-region radius in the internal iteration \( t_k \). The fact that \( \|g_k\| \geq \varepsilon > 0 \), (H2), and (17) imply
\[
q_k(0) - q_k(d_{tk}^k) \geq \beta \|g_k\| \min \left\{ \delta_{tk}^k, \frac{\|g_k\|}{\|B_k\|} \right\} \geq \beta \varepsilon \min \left\{ \delta_{tk}^k, \frac{\varepsilon}{M} \right\}.
\]

Then Line 8 of Algorithm 2 implies
\[
\lim_{t_k \to \infty} \delta_{tk}^k = 0.
\]

From This, Lemma 2, and (23), we obtain
\[
|r_k - 1| = \left| \frac{f_k - f(x_k + d_{tk}^k)}{q_k(0) - q_k(d_{tk}^k)} - 1 \right| = \left| \frac{f_k - f(x_k + d_{tk}^k) - (q_k(0) - q_k(d_{tk}^k))}{q_k(0) - q_k(d_{tk}^k)} \right| \\
\leq \frac{O(\|d_{tk}^k\|^2)}{\beta \varepsilon \min \left\{ \frac{\delta_{tk}^k}{\|B_k\|}, \frac{\varepsilon}{M} \right\}} \leq \frac{O((\delta_{tk}^k)^2)}{\beta \varepsilon \min \left\{ \delta_{tk}^k, \frac{\varepsilon}{M} \right\}} \to 0 \quad (t_k \to \infty),
\]

implying that there exists a positive integer \( k_0 \) such that for \( k \geq k_0 \) we have \( r_k \geq \mu_1 \). This and (18) lead to
\[
\hat{r}_k = \frac{T_k - f(x_k + d_{tk}^k)}{q_k(0) - q_k(d_{tk}^k)} \geq \frac{f_k - f(x_k + d_{tk}^k)}{q_k(0) - q_k(d_{tk}^k)} \geq \mu_1,
\]

implying the inner cycle is left after a finite number of inner iterations, i.e., Algorithm 2 is well-defined.

(ii) Assume that there exists a positive integer \( k \) such that for an arbitrary positive integer \( p \) the point \( x_{k+p+1} \) is not very successful. Hence, for any constant \( p = 0, 1, 2, \ldots \), we get
\[
\hat{r}_{k+p} < \mu_2.
\]
The fact that \( \|g_k\| \geq \varepsilon > 0 \), (H2), and (17) imply

\[
T_{k+p} - f(x_{k+p} + d_{k+p}) \geq \mu_1(q_{k+p}(0) - q_{k+p}(d_{k+p})) \\
\geq \beta \mu_1 \|g_{k+p}\| \min \left\{ \delta_{k+p}, \frac{\|g_{k+p}\|}{\|B_{k+p}\|} \right\} \\
\geq \beta \mu_1 \varepsilon \min \left\{ \delta_{k+p}, \frac{\varepsilon}{M} \right\}.
\tag{24}
\]

By using (22) and (24), we can write

\[
\lim_{p \to \infty} \delta_{k+p} = 0.
\tag{25}
\]

From Lemma 2, (25), and (23), we obtain

\[
|r_{k+p} - 1| = \left| \frac{f(x_{k+p}) - f(x_{k+p} + d_{k+p})}{q_{k+p}(0) - q_{k+p}(d_{k+p})} - 1 \right| \\
= \left| \frac{f(x_{k+p}) - f(x_{k+p} + d_{k+p}) - (q_{k+p}(0) - q_{k+p}(d_{k+p}))}{q_{k+p}(0) - q_{k+p}(d_{k+p})} \right| \\
\leq \frac{O(\|d_{k+p}\|^2)}{\beta \varepsilon \min \left\{ \delta_{k+p}, \varepsilon/M \right\}} \leq \frac{O(\delta_{k+p}^2)}{\beta \varepsilon \min \left\{ \delta_{k+p}, \varepsilon/M \right\}} \to 0 \quad (p \to \infty).
\]

Then, for a sufficiently large \( p \), we get \( r_{k+p} \geq \mu_2 \) leading to

\[
\frac{T_{k+p} - f(x_{k+p} + d_{k+p})}{q_{k+p}(0) - q_{k+p}(d_{k+p})} \geq \frac{f(x_{k+p}) - f(x_{k+p} + d_{k+p})}{q_{k+p}(0) - q_{k+p}(d_{k+p})} \geq \mu_2.
\]

implying \( \hat{r}_{k+p} \geq \mu_2 \), for a sufficiently large \( p \). This contradicts with assumption \( \hat{r}_{k+p} < \mu_2 \) giving the result. \( \square \)

Lemma 9(i) implies that the inner cycle will be left after a finite number of internal iterations, and Lemma 9(ii) implies that if the current iteration is not a first-order stationary point, then at least there exists a very successful iteration point, i.e., the trust-region radius \( \delta_k \) can be enlarged. The next result gives the global convergence of the sequence \( \{x_k\} \) of Algorithm 2.

**Theorem 10** Suppose that (H1) and (H2) hold, and suppose the sequence \( \{x_k\} \) is generated by Algorithm 2. Then

\[
\lim \inf_{k \to \infty} \|g_k\| = 0.
\tag{26}
\]

**Proof** We consider two cases: (i) Algorithm 2 has finitely many very successful iterations; (ii) Algorithm 2 has infinitely many very successful iterations.
In Case (i), we suppose that $k_0$ be the largest index of very successful iterations. If $\|g_{k_0+1}\| > 0$, then Lemma 9(ii) implies that there exists a very successful iteration with larger index than $k_0$. This is a contradiction to the definition of $k_0$.

In Case (ii), by contradiction, we assume that there exist constants $\varepsilon > 0$ and $K > 0$ such that

$$\|g_k\| \geq \varepsilon,$$

for all $k \geq K$. If $x_{k+1}$ is a successful iteration and $k \geq K$, then by using (H2), (17), and (27), we get

$$T_k - f(x_k + d_k) \geq \mu_1 (q_k(0) - q_k(d_k))$$

$$\geq \beta \mu_1 \|g_k\| \min \left\{ \delta_k, \frac{\|g_k\|}{\|B_k\|} \right\} \geq \beta \mu_1 \varepsilon \min \left\{ \delta_k, \frac{\varepsilon}{M} \right\} \geq 0. \quad (28)$$

It follows from this inequality and (22) that

$$\lim_{k \to \infty} \delta_k = 0. \quad (29)$$

Then (H2), (17), and (27) imply

$$|r_k - 1| = \left| \frac{f_k - f(x_k + d_k)}{q_k(0) - q_k(d_k)} - 1 \right| = \left| \frac{f_k - f(x_k + d_k) - (q_k(0) - q_k(d_k))}{q_k(0) - q_k(d_k)} \right|$$

$$\leq \frac{O(\|d_k\|^2)}{\beta \varepsilon \min \{\delta_k, \varepsilon / M\}} = \frac{O(\delta^2)}{\beta \varepsilon \min \{\delta_k, \varepsilon / M\}} \to 0 \quad (k \to \infty).$$

This clearly implies that there exists a positive integer $k_1$ such that for $k \geq k_1$ we have

$$\hat{r}_k = \frac{T_k - f(x_k + d_k)}{q_k(0) - q_k(d_k)} \geq \frac{f_k - f(x_k + d_k)}{q_k(0) - q_k(d_k)} \geq \mu_1,$$

implying $x_{k+1} = x_k + d_k$ for $k \geq k_1$, i.e., the trust-region radius remains fixed ($\mu_1 \leq \hat{r}_k < \mu_2$) or will be increased ($\hat{r}_k \geq \mu_2$). Now we construct a subsequence of very successful iterations for $k \geq k_1$. Form Lemma 9(ii) there exists an integer $p_1$ such that $x_{k_1+p_1}$ is very successful. Then we set $i_1 := k_1 + p_1$. Now, from Lemma 9(ii), there exists an integer $p_2$ such that $x_{i_1+p_2}$ is very successful, and so we set $i_2 := i_1 + p_2$. By continuing this procedure we get the index set

$$\mathcal{I} := \{i_1, i_2, i_3 := i_2 + p_3, \ldots \}$$

such that $\{x_{\mathcal{I}}\}$ is the subsequence of very successful iterations in which the trust-region radius is increased. Hence, for $k \geq k_1$, the trust-region radius is fixed or increased in each iteration, which is a contradiction with (29). This implies the result is valid. \qed
Theorem 11 Suppose that (H1) and (H2) hold, and the sequence \( \{x_k\} \) is generated by Algorithm 2. Then
\[
\lim_{k \to \infty} \|g_k\| = 0. \tag{30}
\]

Proof By contradiction, we assume \( \lim_{k \to \infty} \|g_k\| \neq 0 \). Hence there exists \( \varepsilon > 0 \) and an infinite subsequence of \( \{x_k\} \), indexed by \( \{t_i\} \), such that
\[
\|g_{t_i}\| \geq 2\varepsilon > 0, \tag{31}
\]
for all \( i \in \mathbb{N} \). Theorem 10 ensures the existence, for each \( t_i \), a first successful iteration \( r(t_i) > t_i \) such that \( \|g_{r(t_i)}\| < \varepsilon \). We denote \( r_i = r(t_i) \). Hence there exists another subsequence, indexed by \( \{r_i\} \), such that
\[
\|g_k\| \geq \varepsilon \text{ for } t_i \leq k < r_i, \quad \|g_{r_i}\| < \varepsilon. \tag{32}
\]
We now restrict our attention to the sequence of successful iterations whose indices are in the set
\[
\kappa = \{k \in \mathbb{N} | t_i \leq k < r_i\}.
\]
Using (32), for every \( k \in \kappa \), (28) holds. It follows from (22) and (28) that
\[
\lim_{k \to \infty} \delta_k = 0, \tag{33}
\]
for \( k \in \kappa \). Now, using (H2), (17), and \( \|g_k\| \geq \varepsilon \), the condition (23) holds, for \( k \in \kappa \). This, Lemma 2, and (33) lead to
\[
|r_k - 1| = \left| \frac{f_k - f(x_k + d_k)}{q_k(0) - q_k(d_k)} - 1 \right| = \left| \frac{f_k - f(x_k + d_k) - (q_k(0) - q_k(d_k))}{q_k(0) - q_k(d_k)} \right| \\
\leq \frac{O(\|d_k\|^2)}{\beta \varepsilon \min \{\delta_k, \varepsilon / M\}} \leq \frac{O(\delta_k^2)}{\beta \varepsilon \delta_k} \to 0 \quad (k \to \infty, \ k \in \kappa).
\]
Thus, for a sufficiently large \( k + 1 \in \kappa \), we get
\[
f_k - f(x_k + d_k) \geq \mu_1(q_k(0) - q_k(d_k)) \\
\geq \beta \mu_1 \|g_k\| \min \left\{\delta_k, \frac{\|g_k\|}{\|B_k\|}\right\} \geq \beta \mu_1 \varepsilon \min \left\{\delta_k, \frac{\varepsilon}{M}\right\}. \tag{34}
\]
The condition (33) implies that \( \delta_k \leq \varepsilon / M \). Hence, for a sufficiently large \( k \in \kappa \), we obtain
\[
\delta_k \leq \frac{1}{\beta \mu_1} (f_k - f_{k+1}). \tag{35}
\]
Then (18) and (35) imply

$$\|x_t - x_r\| \leq \sum_{j \in \kappa, j = t_i} \|x_j - x_{j+1}\| \leq \sum_{j \in \kappa, j = t_i} \delta_j \leq \frac{1}{\beta \mu_1} (f_{t_i} - f_{r_i}) \leq \frac{1}{\beta \mu_1} (T_{t_i} - f_{r_i}),$$

(36)

for a sufficiently large $i$. Now, Corollary 8 implies

$$0 \leq \lim_{i \to \infty} \|x_t - x_r\| \leq \lim_{i \to \infty} \frac{1}{\beta \mu_1} (T_{t_i} - f_{r_i}) = 0,$$

leading to

$$\lim_{i \to \infty} \|x_t - x_r\| = 0.$$

Since the gradient is continuous, we get

$$\lim_{i \to \infty} \|g_{t_i} - g_{r_i}\| = 0.$$ 

(37)

In view of the definitions of $\{t_i\}$ and $\{r_i\}$, it is impossible, guaranteeing $\|g_{t_i} - g_{r_i}\| \geq \varepsilon$. Therefore, there is no subsequence that satisfies (31) giving the result.

Note that, in view of the theorem in Page 709 of [28], there is no limit point of the sequence $\{x_k\}$ to be a local maximizer of $f$.

The next result gives the global convergence of the sequence generated by Algorithm 2 to second-order stationary points. To this end, similar to [16], an additional assumption is needed:

(\textbf{H4}) If $\lambda_{\min}(B_k)$ represents the smallest eigenvalue of the symmetric matrix $B_k$, then there exists a positive scalar $c_3$ such that

$$q_k(0) - q_k(d_k) \geq c_3 \lambda_{\min}(B_k) \delta^2.$$

\textbf{Theorem 12} Suppose that $f$ is twice continuously differentiable and also suppose that (H1)-(H4) hold. Then there exists a limit point $x^*$ of the sequence $\{x_k\}$ generated by Algorithm 2 such that $\nabla^2 f(x^*)$ is positive semidefinite.

\textit{Proof} The proof is similar to Theorem 3.4 in [16].

The next two results show that Algorithm 2 can be reduced to quasi-Newton or Newton methods, where the sequence $\{x_k\}$ generated by these schemes can attain local superlinear and quadratic convergence rates under some conditions, respectively.

\textbf{Theorem 13} Suppose that (H1)-(H3) hold, and also suppose that the sequence $\{x_k\}$ generated by Algorithm 2 converges to $x^*$, $\|d_k\| = \| - B_k^{-1} g_k \| \leq \delta_k$, $H(x) = \nabla^2 f(x)$ is continuous in a neighborhood $N(x^*, \varepsilon)$ of $x^*$, and $B_k$ satisfies

$$\lim_{k \to \infty} \left\| \frac{[B_k - H(x^*)]d_k}{\|d_k\|} \right\| = 0.$$ 

(38)
then

(i) There exists a constant \( k_1 \) such that for all \( k \geq k_1 \) we have \( x_{k+1} = x_k + d_k \);

(ii) The sequence \( \{x_k\} \) generated by Algorithm 2 converges to \( x^* \) superlinearly.

Proof (i) The condition (38) implies

\[
\lim_{k \to \infty} \frac{\| g_k + H(x^*)d_k \|}{\|d_k\|} = 0,
\]

leading to

\[
d_k = -H(x^*)^{-1}g_k + o(\|d_k\|).
\]

This implies that

\[
\|d_k\| \leq \|H(x^*)^{-1}\| \|g_k\| + o(\|d_k\|).
\]

Theorem 11 implies that \( \|g_k\| \to 0 \) as \( k \to \infty \). This and (40) give

\[
\lim_{k \to \infty} \|d_k\| = 0.
\]

This, (18), and (H2) imply

\[
|r_k - 1| = \left| \frac{f_k - f(x_k + d_k)}{q_k(0) - q_k(d_k)} - 1 \right| = \left| \frac{f_k - f(x_k + d_k) - (q_k(0) - q_k(d_k))}{q_k(0) - q_k(d_k)} \right| \\
\leq \frac{O(\|d_k\|^2)}{\beta \varepsilon \min \{\delta_k, \varepsilon/M\}} \leq \frac{O(\|d_k\|^2)}{\beta \varepsilon \min \{\|d_k\|, \varepsilon/M\}} \to 0 \quad (k \to \infty).
\]

This clearly implies that there exists a positive integer \( k_1 \) such that for \( k \geq k_1 \) we have \( x_{k+1} = x_k + d_k \).

(ii) From \( d_k = -B_k^{-1}g_k \), we obtain

\[
\frac{\| g_k + H_k d_k \|}{\|d_k\|} = \frac{\| [H_k - B_k]d_k \|}{\|d_k\|} \leq \frac{\| [H_k - H(x^*)]d_k \|}{\|d_k\|} + \frac{\| B_k - H(x^*) \|d_k \|}{\|d_k\|}.
\]

This and (30) lead to

\[
\lim_{k \to \infty} \frac{\| g_k + H_k d_k \|}{\|d_k\|} = 0.
\]

Now Theorem 3.6 in [33] implies that \( \{x_k\} \) generated by Algorithm 2 converges to \( x^* \) superlinearly. \( \square \)

Notice that if \( f \) is thrice continuously differentiable and the upper level set \( L(x_0) \) is bounded, then (H1) implies that \( \|\nabla^3 f(x)\| \) is uniformly continuous and bounded on the open bounded convex set \( \Omega \) involving \( L(x_0) \). Hence, by using the mean value theorem, there exists a constant \( L > 0 \) such that \( \|\nabla^3 f(x)\| \leq L \) implying

\[
\| H(x) - H(y) \| \leq L\| x - y \|,
\]

\( \square \) Springer
for all \( x, y \in \Omega \). This implies that Hessian of \( f \) is Lipschitz continuous. This condition can guarantee the quadratic convergence of the sequence \( \{x_k\} \) generated by Algorithm 2. The details are summarized in the next result.

**Theorem 14** Suppose that \( f(x) \) is a twice continuously differentiable function on \( \mathbb{R}^n \), and all assumptions of Theorem 11 hold. If \( \|d_k\| = \|-H_k^{-1}g_k\| \leq \delta_k \), and there exists a neighborhood \( N(x^*, \epsilon) \) of \( x^* \) such that \( H(x) \) is Lipschitz continuous on \( N(x^*, \epsilon) \), i.e., there exists \( L \) such that

\[
\|H(x) - H(y)\| \leq L \|x - y\|,
\]

then

(i) There exists a constant \( k_2 \) such that for all \( k \geq k_2 \) we have \( x_{k+1} = x_k + d_k \);

(ii) The sequence \( \{x_k\} \) generated by Algorithm 2 converges to \( x^* \) quadratically.

**Proof** (i) By replacing \( B_k \) by \( H_k \) in Theorem 13, we obtain that there exists an integer \( k_2 > 0 \) such that

\[
x_{k+1} = x_k - H_k^{-1}g_k,
\]

for all \( k \geq k_1 \).

(ii) The condition described in (i) and Theorem 3.5 in [33] give the results.

\( \square \)

### 4 Numerical experiments

In this section we report numerical results for Algorithm 2 equipped with two novel nonmonotone terms proposed in Sect. 2 for solving unconstrained optimization problems. In our experiments we use several version of Algorithm 2 employing state-of-the-art nonmonotone terms. In details, we consider

- NMTR-G: Algorithm 2 with the nonmonotone term of Grippo et al. [28];
- NMTR-H: Algorithm 2 with the nonmonotone term of Zhang and Hager [47];
- NMTR-N: Algorithm 2 with the nonmonotone term of Amini et al. [8];
- NMTR-M: Algorithm 2 with the nonmonotone term of Ahookhosh et al. [3];
- NMTR-1: Algorithm 2 with the nonmonotone term (14);
- NMTR-2: Algorithm 2 with the nonmonotone term (15).

In the experiments we used 112 test problems of the CUTEst test collections [25] from dimension 2 to 5000, where we ignore test problems with the dimension greater than 5000. All of the codes are written in MATLAB using the same subroutine, and they are tested on 2Hz core i5 processor laptop with 4GB of RAM with the double-precision data type. The initial points are standard ones proposed in CUTEst. All the algorithms use the radius

\( \copyright \) Springer
\[
\delta_{k+1} = \begin{cases} 
c_1 \|d_k\| & \text{if } \hat{r}_k < \mu_1, \\
\delta_k & \text{if } \mu_1 \leq \hat{r}_k < \mu_2, \\
\max\{\delta_k, c_2 \|d_k\|\} & \text{if } \hat{r}_k \geq \mu_2, 
\end{cases}
\]

where
\[
\mu_1 = 0.05, \quad \mu_2 = 0.9, \quad c_1 = 0.25, \quad c_2 = 2.5, \quad \delta_0 = 0.1\|g_k\|.
\]

see [27]. In the model \(q_k(2)\), an approximation for Hessian is generated by the BFGS updating formula
\[
B_{k+1} = B_k + \frac{y_k y_k^T}{s_k^T y_k} - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k},
\]
where \(s_k = x_{k+1} - x_k\) and \(y_k = g_{k+1} - g_k\). For NMTR-G, NMTR-N, NMTR-1 and NMTR-2, we set \(N = 10\). As discussed in [47], NMLS-H uses \(\eta_k = 0.85\). On the basis of our experiments, we update the parameter \(\eta_k\) by
\[
\eta_k = \begin{cases} 
\eta_0/2 & \text{if } k = 1, \\
(\eta_{k-1} + \eta_{k-2})/2 & \text{if } k \geq 2,
\end{cases}
\]
for NMTR-N, NMTR-M, NMTR-1 and NMTR-2, where the parameter \(\eta_0\) will be tuned to get a better performance. To solve the quadratic subproblem (3), we use the Steihaug-Toint scheme [14] (Chapter 7, Page 205) where the scheme is terminated if
\[
\|g(x_k + d)\| \leq \min\left\{\frac{1}{10}, \|g_k\|^{1/2}\right\} \|g_k\| \quad \text{or} \quad \|d\| = \delta_k.
\]
In our experiments the algorithms are stopped whenever the total number of iterations exceeds 10,000 or
\[
\|g_k\| < \varepsilon
\]
holds with the accuracy parameter \(\varepsilon = 10^{-5}\).

To compare the results appropriately, we use the performance profiles of Dolan and Moré in [19], where the measures of performance are the number of iterations \((N_i)\), the number of function evaluations \((N_f)\), and the number of gradient evaluations \((N_g)\). In the algorithms considered, the number of iterations and gradient evaluations are the same, so we only consider the performance of gradients. It is believed that the computational cost of a gradient is as much as the computational cost three function values, i.e., we further consider the measure \(N_f + 3N_g\). The performance of each code is measured by considering the ratio of its computational outcome versus the best numerical outcome of all codes. This profile offers a tool for comparing the performance of iterative schemes in a statistical structure. Let \(S\) be a set of all algorithms and \(P\) be a set of test problems. For each problem \(p\) and solver \(s\), \(t_{p,s}\) is the computational outcome regarding to the performance index, which is used in defining the performance ratio
\[
r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : s \in S\}}.
\]
If an algorithm $s$ is failed to solve a problem $p$, the procedure sets $r_{p,s} = r_{\text{failed}}$, where $r_{\text{failed}}$ should be strictly larger than any performance ratio (46). For any factor $\tau$, the overall performance of an algorithm $s$ is given by

$$\rho_s(\tau) = \frac{1}{n_P} \text{size}\{p \in \mathcal{P} : r_{p,s} \leq \tau\}.$$ 

In fact $\rho_s(\tau)$ is the probability that a performance ratio $r_{p,s}$ of the algorithm $s \in S$ is within a factor $\tau \in \mathbb{R}^n$ of the best possible ratio. The function $\rho_s(\tau)$ is a distribution function for the performance ratio. In particular, $\rho_s(1)$ gives the probability that the algorithm $s$ wins over all other considered algorithms, and $\lim_{\tau \to r_{\text{failed}}} \rho_s(\tau)$ gives the probability of that the algorithm $s$ solve all considered problems. Hence the performance profile can be considered as a measure of efficiency for comparing iterative schemes. In Figs. 3 and 4, the x-axis shows the number $\tau$ while the y-axis inhibits $P(r_{p,s} \leq \tau : 1 \leq s \leq n_s)$.

4.1 Experiments with highly nonlinear problems

In this subsection we give some numerical results regarding the implementation of NMTR-1 and NMTR-2 compared with TTR on some two-dimensional highly nonlinear problems involving a curved narrow valley. More precisely, we consider the Nesterov–Chebyshev–Rosenbrock, Maratos, and NONDIA functions, see, for example, [9]. In Example 1 the Nesterov–Chebyshev–Rosenbrock function is given, and the Maratos and NONDIA functions are given by

$$f(x_1, x_2) = x_1 + \theta_1(x_1^2 + x_2^2 - 1)^2 \quad \text{(Maratos function)}$$

and

$$f(x_1, x_2) = (1 - x_2)^2 + \theta_2(x_1 - x_2)^2 \quad \text{(NONDIA function)},$$

respectively, where we consider $\theta_1 = 10$ and $\theta_2 = 100$.

We solve the problem (1) for these three functions using TTR, NMTR-1, and NMTR-2, and the results regarding the number of iterations and function evaluations are summarized in Table 1. To give a clear view of the behaviour of TTR, NMTR-1, and NMTR-2, we depict the contour plot of the considered functions and iterations obtained by the algorithms in Fig. 2a, c, and e. In all three cases, one can see that

| Problem name     | Dimension | Initial point   | TTR | NMTR-1 | NMTR-2 |
|------------------|-----------|----------------|-----|--------|--------|
|                  |           |                | $N_g$ | $N_f$  | $N_g$  | $N_f$  |
| Nes–Cheb–Rosen   | 2         | $(-1, 1.5)$    | 32   | 41     | 27     | 34     | 22     | 29     |
| Maratos          | 2         | $(1, 0.95)$    | 31   | 40     | 24     | 29     | 22     | 29     |
| NONDIA           | 2         | $(-0.9, 1.17)$ | 24   | 34     | 27     | 34     | 11     | 17     |
Two globally convergent nonmonotone trust-region methods...

Fig. 2 A comparison among NMTR-1, MNTR-2, and TTR: a, c, and e respectively illustrate the contour plots of the two-dimensional Nesterov–Chebyshev–Rosenbrock, Maratos, and NONDIA functions and iterations of NMTR-1, MNTR-2, and TTR; b, d, and f show the diagram of function values versus iterations.
Fig. 3 Performance profiles of NMTR-1 and NMTR-2 with the performance measures $N_g$, $N_f$, and $N_f + 3N_g$: a and b display the number of iterations ($N_f$) or gradient evaluations ($N_g$); c and d show the number of function evaluations ($N_f$); e and f display the hybrid measure $N_f + 3N_g$.
NMTR-1 and NMTR-2 need less iterations and function values compared with TTR to solve the problem. Moreover, TTR behaves monotonically and follows the bottom of the associated valley, while NMTR-1 and NMTR-2 fluctuated in the valley.

4.2 Experiments with CUTEst test problems

In this subsection we give numerical results regarding experiments with NMTR-1 and NMTR-2 on the CUTEst test problems (Table 2) compared with NMTR-G, NMTR-H, NMTR-N, and NMTR-M.

To get a better performance from NMTR-1 and NMTR-2, we tune the parameter $\eta_0$ by testing several fixed values of $\eta_0$ for both algorithms, where we use $\eta_0 = 0.15, 0.25, 0.35, 0.45$. The corresponding versions of the algorithms NMTR-1 and NMTR-2 are denoted by NMTR-1-0.15, NMTR-1-0.25, NMTR-1-0.35,
NMTR-1-0.45, NMTR-2-0.15, NMTR-2-0.25, NMTR-2-0.35, and NMTR-2-0.45, respectively. The results are summarized in Fig. 3 for three measures: the number function evaluations; the number gradient evaluations; the mixed measure $N_f + 3N_g$.

In Fig. 3a, c and e illustrate that the results of NMTR-1, where it produces the best results with $\eta_0 = 0.25$. From Fig. 3b, d, and f, it can be seen that the best results are produced by $\eta_0 = 0.45$. Hence for NMTR-1 we use $\eta_0 = 0.25$ and for NMTR-2 use $\eta_0 = 0.45$ in the remainder of our experiments.

We here test NMTR-G, NMTR-H, NMTR-N, NMTR-M, NMTR-1, and NMTR-2 for solving the unconstrained problem (1) and compare the produced results. We illustrate the results of comparison in Fig. 4 by using performance profiles for the measures $N_g$, $N_f$, and $N_f + 3N_g$. In Fig. 4a displays for the number of gradient evaluations, where the best results attained by NMTR-2 and then by NMTR-N with about 63 and 52% of the most wins, respectively. NMTR-1 is comparable with NMTR-G, NMTR-H, NMTR-N, but its diagram grows up faster than the others, which means its performance is close to the performance of the best method NMTR-2. Figure 4b shows for the number of function evaluations and has a similar interpretation of Fig. 4a, however, NMTR-2 attains about 60% of the most wins. In Fig. 4c and d display for the mixed measure $N_f + 3N_g$ with $\tau = 1.5$ and $\tau = 5.5$, respectively. In this case NMTR-2 outperforms the others by attaining about 58% of the most wins, and the others have comparable results, however, the diagrams of NMTR-1 and NMTR-M grow up faster than the others implying that they perform close to the best algorithm NMTR-2.

5 Concluding remarks

In this paper we give some motivation for employing nonmonotone strategies in trust-region frameworks. Then we introduce two new nonmonotone terms and combine them into the traditional trust-region framework. It is shown that the proposed methods are globally convergent to first- and second-order stationary points. Moreover local superlinear and quadratic convergence are established. Applying these methods on some highly nonlinear test problems involving a curved narrow valley show that they have a promising behaviour compared with the monotone trust-region method. Numerical experiments on a set of test problems from the CUTEst test collection show the efficiency of the proposed nonmonotone methods.

Further research can be done in several aspects. For example, by combining the proposed nonmonotone trust-region methods with various adaptive radius, more efficient trust-region schemes can be derived, see, for example, [2, 7]. The combination of the proposed nonmonotone terms with several inexact line searches such as Armijo, Wolfe, and Goldstein is also interesting, see [6, 8]. The extension of the proposed method for constrained nonlinear optimization could be interesting, especially for nonnegativity constraints and box constraints, see, for example, [10–12, 34, 42, 43]. It also could be interesting to employ nonmonotone schemes for solving nonlinear least squares and system of nonlinear equations, see [5] and references therein. Moreover, investigating new adaptive formulas for the parameter $\eta_k$ can be precious to improve the computational efficiency.
Acknowledgments  The authors would like to thank three anonymous referees for giving many useful suggestions that improves the paper.

Appendix

See Table 2.

Table 2  The list of test problems and associated dimensions

| Problem name | Dimension | Problem name | Dimension | Problem name | Dimension |
|--------------|-----------|--------------|-----------|--------------|-----------|
| AICRFTB      | 8         | DIXMAANK     | 3000      | MSQRTALS     | 1024      |
| ALLINITU     | 4         | DIXMAANL     | 3000      | MSQRTBL5     | 1024      |
| ARGLINA      | 200       | DJTL         | 2         | NONDQUAR     | 5000      |
| ARGLINB      | 200       | DQDRTIC      | 5000      | OSBORNEA     | 5         |
| ARGLINC      | 200       | DQRTIC       | 5000      | OSBORNEB     | 11        |
| ARWHEAD      | 5000      | EDENSCH      | 2000      | PALMER1C     | 8         |
| BARD         | 3         | EG2          | 1000      | PALMER1D     | 7         |
| BDQRTIC      | 5000      | EIGENALS     | 2550      | PALMER2C     | 8         |
| BEALE        | 2         | ENGV AL1     | 5000      | PALMER3C     | 8         |
| BIGGS3       | 6         | ENGV AL2     | 3         | PALMER4C     | 8         |
| BIGGS5       | 6         | ERRINROS     | 50        | PALMER5C     | 6         |
| BIGGS6       | 6         | EXPFIT       | 2         | PALMER6C     | 8         |
| BOX2         | 3         | EXTROSNB     | 1000      | PALMER7C     | 8         |
| BOX3         | 3         | GENROSE      | 500       | PALMER8C     | 8         |
| BRKMCC       | 2         | GROWTHLS     | 3         | PARKCH       | 15        |
| BROWNAL      | 200       | GULF         | 3         | PENALTY1     | 1000      |
| BROWNBS      | 2         | HAIRY        | 2         | PENALTY2     | 200       |
| BROWNDE      | 4         | HATFLDD      | 3         | PENALTY3     | 200       |
| BRYBND       | 5000      | HATFLDE      | 3         | QUARTC       | 5000      |
| CHAINWOO     | 4000      | HEART6LS     | 6         | ROSENBR      | 2         |
| CUBE         | 2         | HEART8LS     | 8         | S308         | 2         |
| DECONVU      | 63        | HELIX        | 3         | SENSORS      | 100       |
| DENSCHNA     | 2         | HELOW        | 3         | SINEVAL      | 2         |
| DENSCHNB     | 2         | HILBERTA     | 2         | SINQUAD      | 5000      |
| DENSCHNC     | 2         | HILBERTB     | 10        | SISER        | 2         |
| DENSCHND     | 3         | HIMMELBB     | 2         | SNAIL        | 2         |
| DENSCHNE     | 3         | HIMMELBF     | 4         | STRATEC      | 10        |
| DENSCHNF     | 2         | HIMMELBG     | 2         | TOINTGOR     | 50        |
| DIXMAANA     | 3000      | HIMMELBH     | 2         | TOINTPS      | 50        |
| DIXMAANB     | 3000      | HYDC20LS     | 99        | TOINTQOR     | 50        |
| DIXMAANC     | 3000      | JENSMP       | 2         | VARDIM       | 200       |
Table 2 continued

| Problem name  | Dimension | Problem name  | Dimension | Problem name  | Dimension |
|---------------|-----------|---------------|-----------|---------------|-----------|
| DIXMAAND      | 3000      | KOWOSB        | 4         | VAREIGVL      | 50        |
| DIXMAANE      | 3000      | LIARWHD       | 5000      | VIBRBEAM      | 8         |
| DIXMAANF      | 3000      | LOGHAIRY      | 2         | WATSON        | 12        |
| DIXMAANG      | 3000      | MANCINO       | 100       | YFITU         | 3         |
| DIXMAANH      | 3000      | MARATOSB      | 2         | ZANGWIL2      | 2         |
| DIXMAANI      | 3000      | MEXHAT        | 2         |               |           |
| DIXMAANJ      | 3000      | MEYER3        | 3         |               |           |

References

1. Ahookhosh, M., Amini, K.: An efficient nonmonotone trust-region method for unconstrained optimization. Numer. Algorithms 59(4), 523–540 (2012)
2. Ahookhosh, M., Amini, K.: A nonmonotone trust region method with adaptive radius for unconstrained optimization problems. Comput. Math. Appl. 60(3), 411–422 (2010)
3. Ahookhosh, M., Amini, K., Bahrami, S.: A class of nonmonotone Armijo-type line search method for unconstrained optimization. Optimization 61(4), 387–404 (2012)
4. Ahookhosh, M., Amini, K., Peyghami, M.R.: A nonmonotone trust-region line search method for large-scale unconstrained optimization. Appl. Math. Model. 36(1), 478–487 (2012)
5. Ahookhosh, M., Esmaeili, H., Kimiaei, M.: An effective trust-region-based approach for symmetric nonlinear systems. Int. J. Comput. Math. 90(3), 671–690 (2013)
6. Ahookhosh, M., Ghaderi, S.: On efficiency of nonmonotone Armijo-type line searches. http://arxiv.org/pdf/1408.2675 (2015)
7. Amini, K., Ahookhosh, M.: A hybrid of adjustable trust-region and nonmonotone algorithms for unconstrained optimization. Appl. Math. Model. 38, 2601–2612 (2014)
8. Amini, K., Ahookhosh, M., Nosratipour, H.: An inexact line search approach using modified nonmonotone strategy for unconstrained optimization. Numer. Algorithms 66, 49–78 (2014)
9. Andrei, N.: An unconstrained optimization test functions collection. Adv. Model. Optim. 10(1), 147–161 (2008)
10. Birgin, E.G., Martínez, J.M., Raydan, M.: Nonmonotone spectral projected gradient methods on convex sets. SIAM J. Optim. 10, 1196–1211 (2000)
11. Birgin, E.G., Martínez, J.M., Raydan, M.: Inexact spectral projected gradient methods on convex sets.IMA J. Numer. Anal. 23, 539–559 (2003)
12. Bonnans, J.F., Panier, E., Tits, A., Zhou, J.L.: Avoiding the Maratos effect by means of a nonmonotone line search, II: inequality constrained problems—feasible iterates. SIAM J. Numer. Anal. 29, 1187–1202 (1992)
13. Chamberlain, R.M., Powell, M.J.D., Lemarechal, C., Pedersen, H.C.: The watchdog technique for forcing convergence in algorithm for constrained optimization. Math. Progr. Study 16, 1–17 (1982)
14. Conn, A.R., Gould, N.I.M., Toint, PhL: Trust-Region Methods, Society for Industrial and Applied Mathematics. SIAM, Philadelphia (2000)
15. Davidon, W.C.: Conic approximation and collinear scaling for optimizers. SIAM J. Numer. Anal. 17, 268–281 (1980)
16. Deng, N.Y., Xiao, Y., Zhou, F.J.: Nonmonotonic trust region algorithm. J. Optim. Theory Appl. 76, 259–285 (1993)
17. Dennis, J.E., Li, S.B., Tapia, R.A.: A unified approach to global convergence of trust region methods for nonsmooth optimization. Math. Progr. 68, 319–346 (1995)
18. Di, S., Sun, W.: Trust region method for conic model to solve unconstrained optimization problems. Optim. Methods Softw. 6, 237–263 (1996)
19. Dolan, E., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Progr. 91, 201–213 (2002)
20. Erway, J.B., Gill, P.E.: A subspace minimization method for the trust-region step. SIAM J. Optim. 20(3), 1439–1461 (2009)
21. Erway, J.B., Gill, P.E., Griffin, J.D.: Iterative methods for finding a trust-region step. SIAM J. Optim. 20(2), 1110–1131 (2009)
22. Fletcher, R.: Practical Methods of Optimization, 2nd edn. Wiley, New York (1987)
23. Ferreira, P.S., Karas, E.W., Sachine, M.: A globally convergent trust-region algorithm for unconstrained derivative-free optimization. Comput. Appl. Math. (2014). doi:10.1007/s40314-014-0167-2
24. Grapiglia, G.N., Yuan, J., Yuan, Y.: A derivative-free trust-region algorithm for composite nonsmooth optimization. Comput. Appl. Math. (2014). doi:10.1007/s40314-014-0201-4
25. Gould, N.I.M., Orban, D., Toint, Ph.L.: CUTEst: a constrained and unconstrained testing environment with safe threads for mathematical optimization. Comput. Optim. Appl. (2014). doi:10.1007/s10589-014-9687-3
26. Gould, N.I.M., Lucidi, S., Roma, M., Toint, Ph.L.: Solving the trust-region subproblem using the Lanczos method. SIAM J. Optim. 9(2), 504–525 (1999)
27. Gould, N.I.M., Orban, D., Sartenaer, A., Toint, Ph.L.: Sensitivity of trust-region algorithms to their parameters. Q. J. Oper. Res. 3, 227–241 (2005)
28. Grippo, L., Lampariello, F., Lucidi, S.: A nonmonotone line search technique for Newton’s method. SIAM J. Numer. Anal. 23, 707–716 (1986)
29. Gürbüzbalaban, M., Overton, M.L.: On Nesterov’s nonsmooth Chebyshev–Rosenbrock functions. Nonlinear Anal. 75, 1282–1289 (2012)
30. Hallabi, M.E., Tapia, R.: A global convergence theory for arbitrary norm trust region methods for nonlinear equations. Mathematical Sciences Technical Report, TR 87–25, Rice University, Houston, TX (1987)
31. Mo, J., Liu, C., Yan, S.: A nonmonotone trust region method based on nonincreasing technique of weighted average of the successive function value. J. Comput. Appl. Math. 209, 97–108 (2007)
32. Moré, J.J., Sorensen, D.C.: Computing a trust region step. SIAM J. Sci. Stat. Comput. 4(3), 553–572 (1983)
33. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer, New York (2006)
34. Panier, E.R., Tits, A.L.: Avoiding the Maratos effect by means of a nonmonotone linesearch. SIAM J. Numer. Anal. 28, 1183–1195 (1991)
35. Rojas, M., Sorensen, D.C.: A trust-region approach to the regularization of large-scale discrete forms of ill-posed problems. SIAM J. Sci. Comput. 26, 1842–1860 (2002)
36. Schultz, G.A., Schnabel, R.B., Byrd, R.H.: A family of trust-region-based algorithms for unconstrained minimization with strong global convergence. SIAM J. Numer. Anal. 22, 47–67 (1985)
37. Sorensen, D.C.: The q-superlinear convergence of a collinear scaling algorithm for unconstrained optimization. SIAM J. Numer. Anal. 17, 84–114 (1980)
38. Sorensen, D.C.: Newton’s method with a model trust region modification. SIAM J. Numer. Anal. 19(2), 409–426 (1982)
39. Steihaug, T.: The conjugate gradient method and trust regions in large scale optimization. SIAM J. Numer. Anal. 20(3), 626–637 (1983)
40. Sun, W.: On nonquadratic model optimization methods. Asia Pac. J. Oper. Res. 13, 43–63 (1996)
41. Toint, Ph.L.: An assessment of nonmonotone linesearch technique for unconstrained optimization. SIAM J. Sci. Comput. 17, 725–739 (1996)
42. Toint, Ph.L.: Nonmonotone trust-region algorithm for nonlinear optimization subject to convex constraints. Math. Progr. 77, 69–94 (1997)
43. Ulbrich, M., Ulbrich, S.: Nonmonotone trust region methods for nonlinear equality constrained optimization without a penalty function. Math. Progr. 95, 103–135 (2003)
44. Xiao, Y., Zhou, F.J.: Nonmonotone trust region methods with curvilinear path in unconstrained optimization. Computing 48, 303–317 (1992)
45. Xiao, Y., Chu, E.K.W.: Nonmonotone trust region methods, Technical Report 95/17, Monash University, Clayton, Australia (1995)
46. Yuan, Y.: Conditions for convergence of trust region algorithms for nonsmooth optimization. Math. Progr. 31, 220–228 (1985)
47. Zhang, H.C., Hager, W.W.: A nonmonotone line search technique for unconstrained optimization. SIAM J. Optim. 14(4), 1043–1056 (2004)
48. Zhou, F., Xiao, Y.: A class of nonmonotone stabilization trust region methods. Computing 53(2), 119–136 (1994)