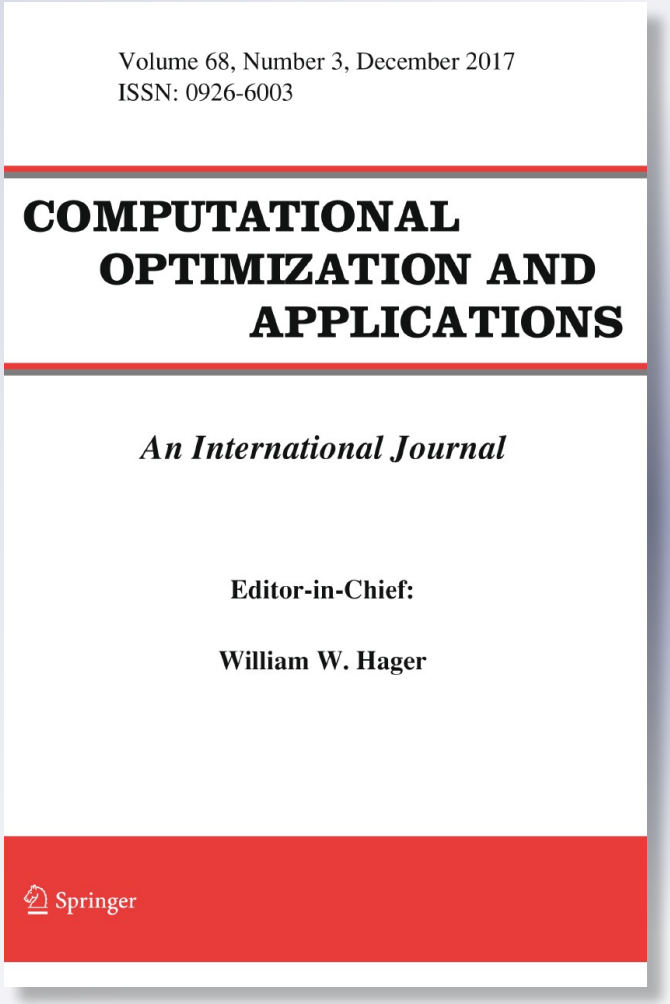
On the worst-case evaluation complexity of non-monotone line search algorithms

Geovani N. Grapiglia & Ekkehard W. Sachs

Computational Optimization and Applications
An International Journal

ISSN 0926-6003
Volume 68
Number 3

Comput Optim Appl (2017) 68:555-577
DOI 10.1007/s10589-017-9928-3



Springer

Comput Optim Appl (2017) 68:555–577
DOI 10.1007/s10589-017-9928-3

On the worst-case evaluation complexity of non-monotone line search algorithms

Geovani N. Grapiglia[1] · **Ekkehard W. Sachs**[2]

Received: 13 July 2016 / Published online: 26 July 2017

Abstract A general class of non-monotone line search algorithms has been proposed by Sachs and Sachs (Control Cybern 40:1059–1075, 2011) for smooth unconstrained optimization, generalizing various non-monotone step size rules such as the modified Armijo rule of Zhang and Hager (SIAM J Optim 14:1043–1056, 2004). In this paper, the worst-case complexity of this class of non-monotone algorithms is studied. The analysis is carried out in the context of non-convex, convex and strongly convex objectives with Lipschitz continuous gradients. Despite de nonmonotonicity in the decrease of function values, the complexity bounds obtained agree in order with the bounds already established for monotone algorithms.

Keywords Nonlinear optimization · Unconstrained optimization · Non-monotone line search · Worst-case complexity

1 Introduction

The worst-case complexity analysis of algorithms for potentially non-convex unconstrained optimization problems has been the subject of a series of recent works. See, for

G. N. Grapiglia was partially supported by CNPq - Brazil, Grant 401288/2014-5.

✉ Geovani N. Grapiglia
grapiglia@ufpr.br

Ekkehard W. Sachs
sachs@uni-trier.de

[1] Departamento de Matemática, Universidade Federal do Paraná, Centro Politécnico, Cx. postal 19.081, 81531-980 Curitiba, Paraná, Brazil

[2] Department of Mathematics, University of Trier, 54286 Trier, Germany

example, Nesterov [22], Nesterov and Polyak [23], Gratton, Sartenaer and Toint [16], Cartis, Gould and Toint [5], Vicente [26], Grapiglia, Yuan and Yuan [13,15], Grapiglia and Nesterov [12], Curtis, Robinson and Samadi [8], Birgin et al. [3], Martínez and Raydan [19], Birgin and Martínez [4] and Dussault [11]. So far, most of these contributions have focused on monotone algorithms, that is, algorithms that do not allow an increase in the values of the objective function in successive iterations, generating in this way a monotonically non-increasing sequence of objective-function values. This property which is important for proving convergence, however, also could lead to relatively short step sizes. This was the motivation in the past to consider non-monotone algorithms, where this monotonicity requirement is relaxed. The way this can be achieved is not unique. In the context of line search algorithms, Grippo et al. [17] proposed the modified Armijo rule

$$f(x_k + \alpha_k d_k) \leq \max_{0 \leq j \leq m(k)} \left[f(x_{k-j}) \right] + \rho \alpha_k \langle \nabla f(x_k), d_k \rangle , \tag{1}$$

where $\alpha_k > 0$ is the step-size, $\rho \in (0, 1)$, $m(0) = 0$ and $0 \leq m(k) \leq \min [m(k-1) + 1, M]$ for a non-negative integer M. A different non-monotone line search strategy, proposed by Zhang and Hager [27], is based on the use of the following modified Armijo rule

$$f(x_k + \alpha_k d_k) \leq C_k + \rho \alpha_k \langle \nabla f(x_k), d_k \rangle , \tag{2}$$

where $C_0 = f(x_0)$,

$$C_{k+1} = \frac{\eta_k Q_k C_k + f(x_{k+1})}{Q_{k+1}}, \tag{3}$$

$Q_0 = 1$, $Q_{k+1} = \eta_k Q_k + 1$ and $\eta_k \in [\eta_{min}, \eta_{max}]$ with $0 \leq \eta_{min} \leq \eta_{max} \leq 1$. Several variants of this technique have been considered in the literature, like the ones developed by Ahookhosh et al. [1] and Mo et al. [20].

If one wants to analyze the complexity of these algorithms one would need to find complexity estimates for each of those methods, requiring a substantial amount of work. See, for example, the contribution done by Cartis, Sampaio and Toint [7], where they studied a class of gradient-related non-monotone algorithms based on rule (1) of Grippo et al. [17]. However, many of these algorithms can be cast into the general framework proposed in [24], where the monotonicity is relaxed by a parameter ν_k indicated in (5). It can be shown that particular choices of ν_k then reduce the general algorithm to the special cases described in [1,20,27]. In this paper we use this advantage to derive worst-case complexity estimates for the general algorithm. This way we also cover the three special cases in one analytic framework. More specifically, when the objective function f is potentially non-convex, we show that the general algorithm take at most $\mathcal{O}(\epsilon^{-2})$ iterations to reduce $\|\nabla f(x_k)\|$ below a user-defined precision $\epsilon \in (0, 1)$. Moreover, when the objective function is convex, we obtain a complexity bound of $\mathcal{O}(\epsilon^{-1})$ iterations to ensure $f(x_k) - f(x^*) \leq \epsilon$, where $f(x^*)$ is the global minimum of f. Furthermore, when the objective function is strongly convex, we show that a subclass of non-monotone algorithms covered in

the general framework, including the algorithm of Zhang and Hager [27], can take at most $\mathcal{O}(\log(\epsilon^{-1}))$ iterations to ensure $f(x_k) - f(x^*) \leq \epsilon$.

The paper is organized as follows. In Sect. 2, we formulate the problem and review some definitions and auxiliary results. In Sect. 3, we analyze the worst-case complexity of the general non-monotone line search algorithm proposed by Sachs and Sachs [24], in the context of non-convex and convex objectives, respectively. In Sect. 4, an improved complexity result is obtained for a subclass of non-monotone algorithms in the case that the objective function is strongly convex. In Sect. 5, we provide a numerical illustration of a new non-monotone approach also covered by our complexity analysis. Finally, in Sect. 6, we summarize the contributions of this work and indicate some directions for future research.

2 Preliminaries

In this section we introduce our notation and give some basic estimates which will be needed in the sequel.

Given a Hilbert space $(X, \langle\,.\,,\,.\,\rangle)$, we consider the minimization problem

$$\min_{x \in X} f(x), \tag{4}$$

where $f : X \to \mathbb{R}$ is Fréchet differentiable. We shall denote the gradient of f at $x \in X$ by $\nabla f(x)$. Thus, $\nabla f(x)$ is the unique vector in X such that

$$f'(x)v = \langle \nabla f(x), v \rangle \quad \text{for all } v \in X.$$

Moreover, given $x_k \in X$, we call $d_k \in X$ a descent direction for x_k if

$$\langle \nabla f(x_k), d_k \rangle < 0.$$

Finally, the norm induced by the inner product $\langle\,.\,,\,.\,\rangle$ will be denoted by $\|\,.\,\|$.

Lemma 1 *Suppose $f : X \to \mathbb{R}$ is Fréchet differentiable and its gradient $\nabla f : X \to X$ is Lipschitz continuous with Lipschitz constant $L > 0$:*

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \quad \forall x, y \in X.$$

Then,

$$|f(y) - f(x) - \langle \nabla f(x), y - x \rangle| \leq \frac{L}{2}\|y - x\|^2 \quad \forall x, y \in X.$$

Proof See, for example, Theorem 1.2.22 in [25]. □

Another estimate bounding the growth of a function as one moves away from the minimum is stated in the next definition.

Definition 1 (*Nesterov and Polyak* [23]) A function $f : X \to \mathbb{R}$ is called gradient dominated of degree $p \in [1, 2]$ if it attains a global minimum $f(x^*)$ at some x^* and for any $x \in X$ we have

$$f(x) - f(x^*) \leq \gamma \|\nabla f(x)\|^p,$$

where γ is a positive constant.

Remark 1 Examples of gradient dominated function are convex functions (with $p = 1$) and strongly convex functions (with $p = 2$). For details, see Sect. 4.2 in [23].

3 Worst-case complexity analysis: general case

For our purpose we use a slight modification of the general framework for algorithms with non-monotone line searches given in Sachs and Sachs [24]. It contains as special cases various versions of non-monotone line searches like the ones developed by Zhang and Hager [27], Mo et al. [20] and Ahookhosh et al. [1].

The following algorithm is a descent algorithm with descent direction d_k allowing for nonmontonocity in the decrease of the function values controlled by a parameter ν_k.

Algorithm 1. (Sachs and Sachs [24])

Step 0 Given $x_0 \in X$, $\alpha_0 > 0$, $\beta, \rho \in (0, 1)$ and $\nu_0 \geq 0$, set $k := 0$.
Step 1 Compute a descent direction $d_k \in X$ for x_k.
Step 2 Find the smallest integer $l_k \geq 0$ such that

$$f(x_k + \alpha_k \beta^{l_k} d_k) \leq f(x_k) + \rho \alpha_k \beta^{l_k} \langle \nabla f(x_k), d_k \rangle + \nu_k. \tag{5}$$

Step 3 Compute $\nu_{k+1} \geq 0$, set $x_{k+1} = x_k + \alpha_k \beta^{l_k} d_k$, $\alpha_{k+1} = \alpha_k \beta^{l_k - 1}$, $k := k + 1$ and go to Step 1.

Our analysis of Algorithm 1 will be conducted under the following assumptions:

A1 The objective function $f : X \to \mathbb{R}$ is Fréchet differentiable and its gradient $\nabla f : X \to X$ is Lipschitz continuous with Lipschitz constant $L > 0$.
A2 For all k,

$$\langle \nabla f(x_k), d_k \rangle \leq -c_1 \|\nabla f(x_k)\|^2 \quad \text{and} \quad \|d_k\| \leq c_2 \|\nabla f(x_k)\|$$

for some constants $c_1, c_2 > 0$.
A3 There exists $f_{low} \in \mathbb{R}$ such that $f(x) \geq f_{low}$ for all $x \in X$.
A4 The nonmonotonicty is controlled by $\sum_{k=0}^{+\infty} \nu_k < +\infty$.

Remark 2 Assumption A2 includes the usual descent direction $d_k = -\nabla f(x_k)$, for which we have $c_1 = c_2 = 1$. Another possible choice is the spectral direction $d_k = -\lambda_k \nabla f(x_k)$ with

$$\lambda_k = \max \left\{ \lambda_{min}, \min \left\{ \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}}, \lambda_{max} \right\} \right\}$$

where $0 < \lambda_{min} < \lambda_{max}$, $s_{k-1} = x_k - x_{k-1}$ and $y_{k-1} = \nabla f(x_k) - \nabla f(x_{k-1})$. In this case, we have $c_1 = \lambda_{min}$ and $c_2 = \lambda_{max}$. Remenber that $s_{k-1}^T s_{k-1}/s_{k-1}^T y_{k-1}$ is the Barzilai–Borwein choice of the step length. For details, see Birgin et al. [2].

We begin the complexity analysis by estimating the step sizes from below in the following lemma.

Lemma 2 *Suppose that A1 and A2 hold. Then, for all k,*

$$\alpha_k \geq \min\left\{\alpha_0, \frac{2(1-\rho)c_1}{Lc_2^2}\right\} \equiv \bar{\alpha}. \tag{6}$$

Proof Clearly the result is true for $k = 0$. Suppose that (6) holds for some $k \geq 0$. We shall prove by induction that (6) also holds for $k + 1$. In fact, if $l_k = 0$, then by the induction assumption

$$\alpha_{k+1} = \beta^{-1}\alpha_k \geq \alpha_k \geq \bar{\alpha},$$

that is, (6) holds for $k+1$. Let us now assume that $l_k \geq 1$. In this case, by the definition of α_k we have

$$f(x_k + \alpha_{k+1}d_k) - f(x_k) > \rho\alpha_{k+1}\langle \nabla f(x_k), d_k\rangle + v_k. \tag{7}$$

On the other hand, from A1 and Lemma 1 it follows that

$$f(x_k + \alpha_{k+1}d_k) - f(x_k) \leq \alpha_{k+1}\langle \nabla f(x_k), d_k\rangle + \frac{L\alpha_{k+1}^2}{2}\|d_k\|^2. \tag{8}$$

Thus, combining (7) and (8) we get

$$\rho\langle \nabla f(x_k), d_k\rangle + \frac{v_k}{\alpha_{k+1}} \leq \langle \nabla f(x_k), d_k\rangle + \frac{L\alpha_{k+1}}{2}\|d_k\|^2.$$

Hence we can estimate α_k from below

$$\begin{aligned}\alpha_{k+1} &\geq \frac{2(1-\rho)}{L}\left(-\frac{\langle \nabla f(x_k), d_k\rangle}{\|d_k\|^2}\right) + \frac{2v_k}{L\|d_k\|^2\alpha_{k+1}} \\ &\geq \frac{2(1-\rho)c_1}{Lc_2^2}.\end{aligned}$$

Therefore (6) also holds in this case. □

This result on the step sizes puts us in the position to prove a first complexity estimate.

Theorem 1 *Suppose that A1–A4 hold. For given $\epsilon \in (0, 1]$, Algorithm 1 takes at most*

$$\left\lceil \left(\frac{f(x_0) - f_{low} + \sum_{k=0}^{\infty} v_k}{\kappa_c}\right)\epsilon^{-2} \right\rceil$$

iterations to ensure $\|\nabla f(x)\| \leq \epsilon$, *where*

$$\kappa_c = \min\left\{\rho\beta\alpha_0 c_1, \frac{2\beta\rho(1-\rho)c_1^2}{Lc_2^2}\right\}. \tag{9}$$

Proof It follows from (5), A2, Lemma 2 and $v_k \geq 0$ that

$$\begin{aligned} v_k + f(x_k) - f(x_{k+1}) &\geq \rho\alpha_k\beta^{l_k}\left(-\langle\nabla f(x_k), d_k\rangle\right) \\ &\geq \rho\beta\alpha_{k+1}c_1\|\nabla f(x_k)\|^2 \geq \kappa_c\|\nabla f(x_k)\|^2, \end{aligned} \tag{10}$$

where κ_c is defined in (9). Let $j_1 \leq +\infty$ be the first iteration such that $\|\nabla f(x_{j_1+1})\| \leq \epsilon$. Then,

$$\|\nabla f(x_k)\| > \epsilon \quad \text{for} \quad k = 0, \ldots, j_1. \tag{11}$$

Thus, it follows from (10) and (11) that

$$v_k + f(x_k) - f(x_{k+1}) \geq \kappa_c\epsilon^2 \quad k = 0, \ldots, j_1. \tag{12}$$

Using (12), A3 and A4 we obtain

$$\begin{aligned} (j_1+1)\kappa_c\epsilon^2 &= \sum_{k=0}^{j_1}\kappa_c\epsilon^2 \leq \sum_{k=0}^{j_1}(v_k + f(x_k) - f(x_{k+1})) \\ &= f(x_0) - f(x_{j_1+1}) + \sum_{k=0}^{j_1} v_k \leq f(x_0) - f_{low} + \sum_{k=0}^{\infty} v_k. \end{aligned}$$

Hence the statement of the theorem follows from

$$j_1 + 1 \leq \left(\frac{f(x_0) - f_{low} + \sum_{k=0}^{\infty} v_k}{\kappa_c}\right)\epsilon^{-2}.$$

In particular, by A4 we have $j_1 < +\infty$. □

In the literature [6,9,14,22,23] one can find complexity estimates which are far better than the $\mathcal{O}(\epsilon^{-2})$ result in Theorem 1. However, this comes at the price of a strengthened assumption on the convexity of the objective function. Moreover, we shall consider the following assumption:

A2' For all k, $d_k = -\lambda_k\nabla f(x_k)$, with $0 < \lambda_{min} \leq \lambda_k \leq \lambda_{max} < +\infty$.

Note that A2' implies A2 with $c_1 = \lambda_{min}$ and $c_2 = \lambda_{max}$.

Theorem 2 *Suppose that A1, A2' and A4 hold. Moreover, assume that f is convex and let x^* be a global minimizer of f. Then, for $\epsilon \in (0, 1]$, Algorithm 1 takes at most*

$$\left\lceil\left(\frac{(\beta\bar{\alpha})^{-1}\|x_0 - x^*\|^2 + \rho^{-1}\lambda_{max}^2\left[(f(x_0) - f(x^*)) + \left(\sum_{k=0}^{+\infty} v_k\right)\right]}{2\lambda_{min}}\right)\epsilon^{-1}\right\rceil$$

iterations to ensure $f(x_k) - f(x^*) \leq \epsilon$, *where* $\bar{\alpha}$ *is defined in Lemma 2.*

Proof It follows from (5) and A2' that

$$v_k + f(x_k) - f(x_{k+1}) \geq \rho\beta\lambda_{min}\alpha_{k+1}\|\nabla f(x_k)\|^2.$$

Thus, we have

$$\alpha_{k+1}\|\nabla f(x_k)\|^2 \leq \frac{f(x_k) - f(x_{k+1}) + v_k}{\rho\beta\lambda_{min}}. \tag{13}$$

Suppose that $j_1 \leq +\infty$ is the first index such that

$$f(x_{j_1}) - f(x^*) \leq \epsilon.$$

Then,

$$f(x_k) - f(x^*) > \epsilon, \text{ for } k = 0, \ldots, j_1 - 1. \tag{14}$$

From (13), we have

$$\sum_{k=0}^{j_1-1} \alpha_{k+1}\|\nabla f(x_k)\|^2 \leq \frac{(f(x_0) - f(x^*)) + \left(\sum_{k=0}^{+\infty} v_k\right)}{\rho\beta\lambda_{min}}. \tag{15}$$

On the other hand, from A2' and the convexity of f it follows that it

$$\begin{aligned}\|x_{k+1} - x^*\|^2 &= \|(x_k - x^*) - \alpha_{k+1}\beta\lambda_k\nabla f(x_k)\|^2 \\ &= \|x_k - x^*\|^2 - 2\alpha_{k+1}\lambda_k\beta(x_k - x^*)^T\nabla f(x_k) + \alpha_{k+1}^2\beta^2\lambda_k^2\|\nabla f(x_k)\|^2 \\ &\leq \|x_k - x^*\|^2 - 2\lambda_k\beta\alpha_{k+1}(f(x_k) - f(x^*)) + \beta^2\lambda_k^2\alpha_{k+1}^2\|\nabla f(x_k)\|^2.\end{aligned}$$

Thus, by Lemma 2 and A2',

$$f(x_k) - f(x^*) \leq \frac{1}{2\lambda_{min}\beta\bar{\alpha}}\left(\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\right) + \frac{\lambda_{max}\beta}{2}\alpha_{k+1}\|\nabla f(x_k)\|^2.$$

Summing up these inequalities for $k = 0, \ldots, j_1 - 1$ and using (15) we get

$$\begin{aligned}&\sum_{k=0}^{j_1-1}(f(x_k) - f(x^*)) \\ &\quad\leq \frac{(\beta\bar{\alpha})^{-1}\|x_0 - x^*\|^2 + \rho^{-1}\lambda_{max}\left[(f(x_0) - f(x^*)) + \left(\sum_{k=0}^{+\infty} v_k\right)\right]}{2\lambda_{min}}.\end{aligned} \tag{16}$$

Thus, from (14) and (16), it follows that

$$j_1\epsilon \leq \frac{(\beta\bar{\alpha})^{-1}\|x_0 - x^*\|^2 + \rho^{-1}\lambda_{max}\left[(f(x_0) - f(x^*)) + \left(\sum_{k=0}^{+\infty} v_k\right)\right]}{2\lambda_{min}}.$$

Therefore,

$$j_1 \leq \left(\frac{(\beta\bar{\alpha})^{-1}\|x_0 - x^*\|^2 + \rho^{-1}\lambda_{max}\left[(f(x_0) - f(x^*)) + \left(\sum_{k=0}^{+\infty} \nu_k\right)\right]}{2\lambda_{min}} \right) \epsilon^{-1}.$$

□

Theorems 1 and 2 state that the complexity of Algorithm 1 with a non-monotone step-size rule is of $\mathcal{O}(\epsilon^{-2})$ iterations in the non-convex case, and of $\mathcal{O}(\epsilon^{-1})$ iterations in the convex case, respectively. The constants in the $\mathcal{O}$-term contain the measure for nonmonotonicity $\sum_{k=0}^{\infty} \nu_k$. It is the smallest, if the nonmonotonicity vanishes and a descent in the function value is enforced at every step. However, an advantage of not requiring a monotone behavior of the outer iterations is that the inner iterations should terminate at an earlier stage. The next theorem gives an upper bound on the total number of function evaluations after $k \geq 1$ iterations.

Theorem 3 *Suppose that A1–A2 hold and let N_k be the total number of function evaluations after $k \geq 1$ iterations of Algorithm 1. Then,*

$$N_k \leq 2(k+1) + \frac{1}{\log(\beta)}\left[\log(\bar{\alpha}) - \log(\alpha_0)\right], \tag{17}$$

where $\bar{\alpha}$ is defined in Lemma 2.

Proof From the definition of α_{k+1} we have that

$$\alpha_{k+1} = \beta^{l_k - 1}\alpha_k \Longrightarrow \log(\alpha_{k+1}) = (l_k - 1)\log(\beta) + \log(\alpha_k)$$

$$\Longrightarrow l_k + 1 = 2 + \frac{1}{\log(\beta)}\left[\log(\alpha_{k+1}) - \log(\alpha_k)\right]. \tag{18}$$

Note that, at each iteration Algorithm 1 performs $l_k + 1$ function evaluations at points of the form $x_k + \alpha_k\beta^i d_k$. Thus, the total number of function evaluations after $k \geq 1$ iterations satisfies the relation

$$\begin{aligned} N_k = \sum_{i=0}^{k}(l_i + 1) &\leq 2(k+1) + \frac{1}{\log(\beta)}\left[\log(\alpha_{k+1}) - \log(\alpha_0)\right] \\ &\leq 2(k+1) + \frac{1}{\log(\beta)}\left[\log(\bar{\alpha}) - \log(\alpha_0)\right]. \end{aligned}$$

□

Remark 3 From (17) we see that Algorithm 1 performs in average two function evaluations per iteration.

4 Worst-case complexity analysis: particular case

In Algorithm 1, it is not specified how to choose the sequence $\{\nu_k\}$ satisfying A4. One way is to choose this sequence a priori, i.e., independently of the progress of the algorithm. For example, a possible choice is $\nu_k = ar^k$, with $a > 0$ and $r \in (0, 1)$. In this section, we give a way of defining ν_k as the iteration progress.

Lemma 3 *Let* $\delta_{k+1} \in [0, 1]$. *Then,*

$$(1 - \delta_{k+1})(f(x_k) + \nu_k) + (\delta_{k+1} - 1) f(x_{k+1}) \geq 0.$$

Proof From Steps 1 and 2 of Algorithm 1 we have

$$f(x_{k+1}) \leq f(x_k) + \nu_k$$

which implies the statement by multiplying both sides with $1 - \delta_{k+1}$. □

Let us consider the following algorithm.

Algorithm 2.

Step 0 Given $x_0 \in X$, $\alpha_0 > 0$, $\beta, \rho \in (0, 1)$, set $\nu_0 = 0$ and $k := 0$.
Step 1 Compute a descent direction $d_k \in X$ for x_k.
Step 2 Find the smallest integer $l_k \geq 0$ such that

$$f(x_k + \alpha_k \beta^{l_k} d_k) \leq f(x_k) + \rho \alpha_k \beta^{l_k} \langle \nabla f(x_k), d_k \rangle + \nu_k. \tag{19}$$

Step 3 Set $x_{k+1} = x_k + \alpha_k \beta^{l_k} d_k$ and $\alpha_{k+1} = \alpha_k \beta^{l_k - 1}$.
Step 3 Compute $\delta_{k+1} \in [\delta_{min}, 1]$ for some constant $\delta_{min} \in (0, 1)$, choose

$$0 \leq \nu_{k+1} \leq (1 - \delta_{k+1})(f(x_k) + \nu_k) + (\delta_{k+1} - 1) f(x_{k+1}), \tag{20}$$

set $k := k + 1$ and go to Step 1.

The upper bound (20) can be expressed as

$$\nu_{k+1} \leq (1 - \delta_{k+1})(\nu_k + f(x_k) - f(x_{k+1}))$$

which shows that the difference of consecutive function values enters the upper bound for ν_{k+1}. If we have a large reduction in the function values in one iteration, then the measure of inexactness has to be reduced also more than in case there would be a small reduction.

Remark 4 The choice of ν_{k+1} satisfying (20) is guaranteed by Lemma 3.

Remark 5 Let us have a closer look at the content of rule (20) for choosing ν_{k+1}. The upper bound on it contains two components: First, it has to be less than $(1 - \delta_{min})\nu_k$ which by itself would yield a geometrically convergent sequence of ν_k. However, as

the following theorem shows, this can be relaxed and an additional slack is allowed described by the remaining term

$$(1-\delta_{k+1})(f(x_k)-f(x_{k+1})).$$

The advantage of (20) over an a *a priori* defined measure of inexactness through a geometrically converging sequence of ν_k lies in the fact that (20) also contains terms with the difference in function values. These quantities are different for each function f and hence problem dependent.

The theorem below establishes a worst-case complexity bounds of $\mathcal{O}(\epsilon^{-2})$ and $\mathcal{O}(\epsilon^{-1})$ iterations for Algorithm 2.

Theorem 4 *Suppose that A1–A3 hold. Then, given $\epsilon \in (0,1]$, Algorithm 2 takes at most $\mathcal{O}(\epsilon^{-2})$ iterations to ensure $\|\nabla f(x_k)\| \leq \epsilon$. Moreover, if A1–A2' hold and f is convex with a global minimizer x^*, then Algorithm 2 takes at most $\mathcal{O}(\epsilon^{-1})$ iterations to ensure $f(x_k)-f(x^*) \leq \epsilon$.*

Proof The concept of Algorithm 2 fits into the framework of Algorithm 1. Thus, we shall prove this result by showing that the sequence $\{\nu_k\}$ in Algorithm 2 satisfies the assumption A4. Then, the conclusion will follow directly from Theorems 1 and 2.

In fact, by (20) we have

$$f(x_{k+1})+\nu_{k+1} \leq (1-\delta_{k+1})(f(x_k)+\nu_k)+\delta_{k+1}f(x_{k+1})$$

$$\Longrightarrow \delta_{k+1}\left[f(x_k)+\nu_k-f(x_{k+1})\right] \leq (f(x_k)+\nu_k)-(f(x_{k+1})+\nu_{k+1}).$$

Thus, it follows from A2 and $\nu_0=0$ that, for all $N \in \mathbb{N}$,

$$\begin{aligned}&\sum_{k=0}^{N}\delta_{k+1}\left[f(x_k)+\nu_k-f(x_{k+1})\right] \leq \sum_{k=0}^{N}\left[(f(x_k)+\nu_k)-(f(x_{k+1})+\nu_{k+1})\right]\\ &= f(x_0)-f(x_{N+1})-\nu_{N+1} \leq f(x_0)-f(x_{N+1}) \leq f(x_0)-f_{low}.\end{aligned}$$

Therefore,

$$\sum_{k=0}^{\infty}\delta_{k+1}(f(x_k)+\nu_k-f(x_{k+1})) \leq f(x_0)-f_{low}. \tag{21}$$

Since $\delta_k \geq \delta_{min}$ for all k and $\nu_0=0$, the inequality (20) implies that

$$\sum_{k=0}^{+\infty}\nu_k=\sum_{k=0}^{+\infty}\nu_{k+1} \leq \sum_{k=0}^{+\infty}(1-\delta_{k+1})\left[f(x_k)+\nu_k-f(x_{k+1})\right]$$

and with (21) we obtain the summability of the sequence v_k

$$\begin{aligned}\sum_{k=0}^{\infty} v_k &\leq \sum_{k=0}^{+\infty} \left(\frac{1-\delta_{k+1}}{\delta_{k+1}}\right) \delta_{k+1}\left[f(x_k)+v_k-f(x_{k+1})\right] \\ &\leq \left(\frac{1-\delta_{min}}{\delta_{min}}\right) \sum_{k=0}^{+infty} \delta_{k+1}\left[f(x_k)+v_k-f(x_{k+1})\right] \\ &\leq \left(\frac{1-\delta_{min}}{\delta_{min}}\right)(f(x_0)-f_{low}).\end{aligned}$$

□

Now, similar as in [23], in order to further improve our complexity estimates we shall consider the following assumption:

A3' The objective function f is a gradient dominated function of degree 2, i.e. $f(x)-f(x^*) \leq \gamma \|\nabla f(x)\|^2$.

To obtain a complexity estimate, one can see from the previous assumption that it suffices to have an estimate for the distance from the minimum in terms of the function values. Due to the inexactness in the step size rule this cannot be achieved directly, however, if the terms v_k, are included in the estimate, we obtain a linear rate of convergence for those quantities as outlined in the next theorem.

Theorem 5 *Suppose that A1–A2 and A3' hold, and let the sequences* $\{x_k\}$ *and* $\{v_k\}$ *be generated by Algorithm 2 with* α_k *updated by the rule*

$$\alpha_{k+1} = \min\left\{\alpha_{max}, \alpha_k \beta^{l_k-1}\right\} \tag{22}$$

where $\alpha_{max} \geq \alpha_0$ *is given. Then, for all k,*

$$f(x_{k+1}) - f(x_*) + v_{k+1} \leq \theta(f(x_k) - f(x^*) + v_k) \tag{23}$$

where

$$\theta = 1 - \kappa_c b_2 \delta_{min}, \quad b_2 = \frac{1}{\kappa_c + \gamma b_1^2} \quad \text{and} \quad b_1 = L\beta\alpha_{max}c_2 + 1. \tag{24}$$

Proof By A2 and the update rule for α_k, we have

$$\|x_{k+1} - x_k\| = \beta\alpha_{k+1}\|d_k\| \leq \beta\alpha_{max}c_2\|\nabla f(x_k)\|. \tag{25}$$

Thus, combining A1 and (25) we obtain

$$\begin{aligned}\|\nabla f(x_{k+1})\| &\leq \|\nabla f(x_{k+1}) - \nabla f(x_k)\| + \|\nabla f(x_k)\| \leq L\|x_{k+1} - x_k\| + \|\nabla f(x_k)\| \\ &\leq (L\beta\alpha_{max}c_2 + 1)\|\nabla f(x_k)\| = b_1\|\nabla f(x_k)\|. \end{aligned} \tag{26}$$

By (20) we have

$$\begin{aligned} f(x_{k+1}) - f(x^*) + v_{k+1} &\leq (1-\delta_{k+1})(f(x_k) + v_k) + \delta_{k+1} f(x_{k+1}) - f(x^*) \\ &= (1-\delta_{k+1})(f(x_k) - f(x^*) + v_k) \\ &\quad + \delta_{k+1}(f(x_{k+1}) - f(x^*)). \end{aligned} \tag{27}$$

Suppose that

$$\|\nabla f(x_k)\|^2 \geq b_2(f(x_k) - f(x^*) + v_k). \tag{28}$$

Thus, by (27), (10) and (28) we get

$$\begin{aligned} f(x_{k+1}) - f(x^*) + v_{k+1} &\leq (1-\delta_{k+1})(f(x_k) - f(x^*) + v_k) \\ &\quad + \delta_{k+1}(f(x_k) + v_k - \kappa_c \|\nabla f(x_k)\|^2 - f(x^*)) \\ &= (f(x_k) - f(x^*) + v_k) - \delta_{k+1}\kappa_c \|\nabla f(x_k)\|^2 \\ &\leq (f(x_k) - f(x^*) + v_k) - \delta_{min}\kappa_c \|\nabla f(x_k)\|^2 \\ &\leq (1 - \kappa_c b_2 \delta_{min})(f(x_k) - f(x^*) + v_k) \\ &= \theta(f(x_k) - f(x^*) + v_k), \end{aligned}$$

that is, (23) holds.

Now, suppose that (28) is not true, that is,

$$\|\nabla f(x_k)\|^2 < b_2(f(x_k) - f(x^*) + v_k).$$

Then, A3' and (26) imply that

$$\begin{aligned} f(x_{k+1}) - f(x^*) &\leq \gamma \|\nabla f(x_{k+1})\|^2 \\ &\leq \gamma b_1^2 \|\nabla f(x_k)\|^2 \\ &< \gamma b_1^2 b_2 (f(x_k) - f(x^*) + v_k). \end{aligned} \tag{29}$$

Thus, combining (27) and (29) we obtain with $b_2\kappa_c = 1 - \gamma b_1^2 b_2$ from (24)

$$\begin{aligned} f(x_{k+1}) - f(x^*) + v_{k+1} &< (1-\delta_{k+1})(f(x_k) - f(x^*) + v_k) \\ &\quad + \delta_{k+1}\gamma b_1^2 b_2 (f(x_k) - f(x^*) + v_k) \\ &\leq \left[1 - \delta_{min}\left(1 - \gamma b_1^2 b_2\right)\right](f(x_k) - f(x^*) + v_k) \\ &= (1 - \delta_{min}\kappa_c b_2)(f(x_k) - f(x^*) + v_k) \\ &= \theta(f(x_k) - f(x^*) + v_k), \end{aligned}$$

that is, (23) also holds in this case. □

The previous estimates put us in the position to prove a complexity estimate under the strengthened conditions on the objective function. As one can see, we do not obtain a q-linear rate of convergence for the function values, as mentioned before, however, a r-linear rate of convergence can be shown, which is sufficient to obtain a complexity estimate.

Theorem 6 *Suppose that A1–A2 and A3' hold and let* $\{x_k\}$ *be generated by Algorithm 2. Then, for all* k,

$$f(x_k) - f(x^*) \leq \theta^k (f(x_0) - f(x^*)) \tag{30}$$

where θ *is defined in* (24). *Consequently, given* $\epsilon \in (0, e^{-1})$, *Algorithm 2 takes at most*

$$\left\lceil \frac{|\log(f(x_0) - f(x^*))| + 1}{|\log(\theta)|} \right\rceil \log(\epsilon^{-1}) \tag{31}$$

iterations to ensure $f(x_k) - f(x^*) \leq \epsilon$.

Proof By Theorem 5,

$$f(x_{k+1}) - f(x^*) + v_{k+1} \leq \theta(f(x_k) - f(x^*) + v_k), \quad k \geq 0.$$

Then with $v_0 = 0$ we have $f(x_1) - f(x^*) + v_1 \leq \theta(f(x_0) - f(x^*))$. Hence by induction. if

$$f(x_k) - f(x^*) + v_k \leq \theta^k (f(x_0) - f(x^*))$$

holds, then

$$f(x_{k+1}) - f(x^*) + v_{k+1} \leq \theta(f(x_k) - f(x^*) + v_k) \leq \theta^{k+1}(f(x_0) - f(x^*))$$

As $v_k \geq 0$, it follows that

$$f(x_k) - f(x^*) \leq \theta^k (f(x_0) - f(x^*)), \quad k \geq 1. \tag{32}$$

Suppose that $f(x_0) - f(x^*) > \epsilon$ and let $j_1 \leq +\infty$ be the first iteration such that

$$f(x_{j_1+1}) - f(x^*) \leq \epsilon.$$

Then, for $k = 0, \ldots, j_1$ we have

$$f(x_k) - f(x^*) > \epsilon. \tag{33}$$

Combining (32) and (33) we obtain

$$\begin{aligned}
&\epsilon < \theta^{j_1}(f(x_0) - f(x^*)) \\
&\quad \Longrightarrow -\log(\epsilon) > j_1(-\log(\theta)) - \log(f(x_0) - f(x^*)) \\
&\quad \Longrightarrow \log(\epsilon^{-1}) > j_1|\log(\theta)| - \log(f(x_0) - f(x^*)) \\
&\quad \Longrightarrow j_1|\log(\theta)| < \log(f(x_0) - f(x^*)) + \log(\epsilon^{-1}) \\
&\quad \leq \left(|\log(f(x_0) - f(x^*))| + 1\right)\log(\epsilon^{-1}) \\
&\quad \Longrightarrow j_1 \leq \left(\frac{|\log(f(x_0) - f(x^*))| + 1}{|\log(\theta)|}\right)\log(\epsilon^{-1}).
\end{aligned}$$

□

Consider again Algorithm 2. Let $C_0 = f(x_0)$,

$$R_k = f(x_k) + \nu_k \quad \text{for all} \quad k \geq 0$$

and

$$C_{k+1} = (1 - \delta_{k+1})R_k + \delta_{k+1} f(x_{k+1}) \quad \text{for all} \quad k \geq 0.$$

Then, Algorithm 2 can be reformulated as follows.

Algorithm 3. (Algorithm 2 reformulated)

Step 0 Given $x_0 \in X$, $\alpha_0 > 0$, $\beta, \rho \in (0, 1)$, set $C_0 = f(x_0)$ and $k := 0$.
Step 1 Compute a descent direction $d_k \in X$ for x_k.
Step 2 Choose $R_k \in [f(x_k), C_k]$ and find the smallest integer $l_k \geq 0$ such that

$$f(x_k + \alpha_k \beta^{l_k} d_k) \leq R_k + \rho \alpha_k \beta^{l_k} \langle \nabla f(x_k), d_k \rangle . \tag{34}$$

Step 3 Set $x_{k+1} = x_k + \alpha_k \beta^{l_k} d_k$ and $\alpha_{k+1} = \alpha_k \beta^{l_k - 1}$.
Step 3 Compute $\delta_{k+1} \in [\delta_{min}, 1]$ for some constant $\delta_{min} \in (0, 1))$, set

$$C_{k+1} = (1 - \delta_{k+1})R_k + \delta_{k+1} f(x_{k+1}), \tag{35}$$

$k := k + 1$ and go to Step 1.

If $R_k = C_k$ for all k, then Algorithm 3 can be seen as a line search counterpart of the non-monotone trust-region algorithm proposed by Gu and Mo [18]. Moreover, for a particular choice of the sequence $\{\delta_k\}$, Algorithm 3 also covers the non-monotone line search algorithm proposed by Zhang and Hager [27]. In fact, consider the choices

$$R_k = C_k \quad \text{and} \quad \delta_{k+1} = \frac{1}{\eta_k Q_k + 1} \text{ for all } k \geq 0,$$

where $Q_0 = 1$, $Q_{k+1} = \eta_k Q_k + 1$ and $\eta_k \in [\eta_{min}, \eta_{max}]$, with $0 \leq \eta_{min} \leq \eta_{max} < 1$. In this case, we have

$$f(x_{k+1}) \leq C_k + \rho \alpha_k \langle \nabla f(x_k), d_k \rangle$$

and

$$C_{k+1} = \frac{\eta_k q_k C_k + f(x_{k+1})}{Q_{k+1}}.$$

Moreover,

$$Q_{k+1} = 1 + \sum_{j=0}^{k} \Pi_{i=0}^{j} \eta_{k-i} \leq 1 + \sum_{j=0}^{k} \eta_{max}^{j+1} \leq \sum_{j=0}^{+\infty} \eta_{max}^{j} = \frac{1}{1 - \eta_{max}}$$

$$\Longrightarrow \delta_{k+1} = \frac{1}{Q_{k+1}} \geq 1 - \eta_{max} \equiv \delta_{min}.$$

Consequently, by Theorems 4 these non-monotone line search algorithms take at most $\mathcal{O}(\epsilon^{-2})$ iterations to ensure $\|\nabla f(x_k)\| \leq \epsilon$ in the non-convex case and $\mathcal{O}(\epsilon^{-1})$ iterations to ensure $f(x_k) - f(x^*) \leq \epsilon$ in the convex case. Moreover, if we impose an upper bound α_{max} to α_k, it follows from Theorem 6 that these algorithms take at most $\mathcal{O}(\log(\epsilon^{-1}))$ iterations to ensure $f(x_k) - f(x^*) \leq \epsilon$ when the objective function is gradient dominated of degree 2 (which is the case when f is strongly convex).

5 Illustrative numerical experiments

From a practical point of view, a key point in Algorithm 3 is the choice of δ_{k+1} at each iteration. Numerical results reported in the literature suggest that it can be very advantageous vary δ_{k+1} dynamically: using values closer to 0 when the iterates are far from a minimizer of f, and using values closer to 1 when the iterates are near to a minimizer. In other words, the strategy is to allow a strong non-monotone behavior of $\{f(x_k)\}$ when the iterates are far from a minimizer, and impose a monotone behavior to $\{f(x_k)\}$ when the iterates are near to a minimizer. Usually, this strategy is implemented by choosing $\{\delta_k\}$ *a priori* such that

$$\delta_1 = \delta_{min} \quad \text{and} \quad \lim_{k \to +\infty} \delta_k = \delta_{max} \in (\delta_{min}, 1].$$

We noticed that, another way to do so (which is also covered by our complexity analysis) is to choose δ_{k+1} by the rule

$$\delta_{k+1} = \phi(\|\nabla f(x_{k+1})\|) \tag{36}$$

where $\phi : \mathbb{R}_+ \to [\delta_{min}, 1]$ is a non-increasing function such that

$$\lim_{t \to 0} \phi(t) = \delta_{max} \in (\delta_{min}, 1].$$

For example, if $\delta_{min} = 0.5$ and $\delta_{max} = 1$, one can use

$$\phi(t) = 0.5 + \frac{0.5}{1 + e^{-a(b-t)}}, \tag{37}$$

with $a, b > 0$. The application of rule (36) is potentially interesting, since this rule exploits the information available about x_{k+1}. In order to investigate its numerical performance, we have tested MATLAB implementations of four versions of Algorithm 3 and also an implementation of the non-monotone algorithm in [17]. Specifically, we considered the following codes:

i. the monotone algorithm obtained with the choices $R_k = C_k$ and $\delta_{k+1} = 1$. We shall refer to this code as "M1".
ii. the non-monotone algorithm in [27] with the choices

$$R_k = C_k, \ \delta_{k+1} = \frac{1}{\eta_k Q_k + 1}, \ Q_0 = 1,$$

$$Q_{k+1} = \eta_k Q_k + 1 \text{ and } \eta_k = 0.85.$$

We shall refer to this code as "NM1".

iii. the non-monotone algorithm in [27] with $\eta_k = 0.85/k$, which we shall refer to as "NM2".

iv. the new non-monotone algorithm obtained with the choices $R_k = C_k$ and $\delta_{k+1} = \phi(\|\nabla f(x_{k+1})\|)$, where ϕ is the function defined in (37) with $a = 2$ and $b = 2.14$. We shall refer to this code as "NM3".

v. the non-monotone algorithm in [17] obtained from Algorithm 3 by setting

$$R_k = C_k = \max_{0 \leq j \leq M_k} f(x_{k-j}),$$

where $M_0 = 0$ and $M_k = \min\{M_{k-1} + 1, 10\}$ for $k > 0$.

In all implementations, we consider the parameters $\alpha_0 = 1$ and $\beta = \rho = 0.5$. The search directions were generated as $d_k = -H_k \nabla f(x_k)$, where H_k is computed using the BFGS update whenever it is possible, namely:

$$H_{k+1} = \begin{cases} \left(I - \dfrac{s_k y_k^T}{s_k^T y_k}\right) H_k \left(I - \dfrac{y_k s_k^T}{s_k^T y_k}\right) + \dfrac{s_k s_k^T}{s_k^T y_k}, & \text{if } s_k^T y_k > 0, \\ H_k, & \text{otherwise.} \end{cases}$$

where $H_0 = I$, $s_k = x_{k+1} - x_k$ and $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$. Our stopping criterion was

$$\|\nabla f(x_k)\|_2 \leq 10^{-5}. \tag{38}$$

The codes were applied to a set of 33 non-convex test problems from [21],[1] most of them highly nonlinear. The tests were performed with MATLAB 7.10.0 (R2010a) on a PC with a 2.70 GHz Intel(R) i5 microprocessor. Problems and results are given in Table 1, where "n" represents the number of variables of the problem, "m" represents the number of component functions considered, "IT" represents the number of iterations, "FE" represents the number of function evaluations and an entry "F" indicates that either the maximum number of 500 iterations was reached or the code ran more than two minutes, without satisfying the stopping criterion (38).

From Table 1, it can be seen that for all codes the number of function evaluations is approximately equal to two times the number of iterations. This result is in accordance to Theorem 3 (see Remark 3). To facilitate the comparison between the three codes, we use the performance profiles proposed by Dolan and Moré [10].[2] Figure 1 shows that for our set of test problems NM2 and NM3 are significantly more efficient than M1, NM1 and NM4 regarding the number of iterations, with an advantage for code NM3 which is based on the new rule (36)–(37). These illustrative numerical results suggest

[1] The MATLAB codes of all test problems considered are freely available in the website http://www.mat.univie.ac.at/~neum/glopt/moretest/.

[2] The performance profiles were generated using the MATLAB code **perf.m** freely available in the website http://www.mcs.anl.gov/~more/cops/.

Table 1 Numerical results for problems from the Moré–Garbow–Hillstrom collection [21]

PROBLEM (n, m)	M1		NM1		NM2		NM3		NM4	
	IT	FE	IT	FE	IT	FE	IT	FE	IT	FE
1. Rosenbrock (2, 2)	45	91	135	270	43	86	41	82	50	100
2. Freudenstein-Roth (2, 2)	27	55	126	252	28	56	26	52	125	250
3. Powell badly scaled (2, 2)	F	F	363	726	171	342	178	356	185	370
4. Brown badly scaled (2, 3)	F	F	288	576	57	114	96	192	234	468
5. Beale (2, 3)	17	34	116	234	18	36	18	36	26	53
6. Jennrich–Sampson (2, 10)	36	72	176	352	36	72	46	92	204	408
7. Helical valley (3, 3)	34	69	156	312	34	69	34	68	45	90
8. Bard (3, 15)	20	40	112	224	19	38	20	40	47	94
9. Gaussian (3, 15)	4	8	4	8	4	8	4	8	4	8
10. Meyer (3, 16)	F	F	F	F	F	F	F	F	F	F
11. Gulf (3, 3)	9	18	27	50	9	18	10	20	10	20
12. Box (3, 3)	F	F	142	284	30	60	30	60	81	162
13. Powell singular (4, 4)	F	F	79	158	39	78	35	70	43	86
14. Wood (4, 6)	90	172	245	490	85	170	90	180	149	298
15. Kowalik–Osborne (4, 11)	F	F	83	116	24	48	23	46	28	56
16. Brown-Dennis (4, 20)	F	F	188	376	38	76	50	100	175	350

Table 1 continued

PROBLEM (n, m)	M1		NM1		NM2		NM3		NM4	
	IT	FE	IT	FE	IT	FE	IT	FE	IT	FE
17. Osborne 1 (5, 33)	F	F	F	F	F	F	F	F	F	F
18. Biggs EXP6 (6, 6)	38	76	130	260	39	78	38	76	40	80
19. Osborne 2 (11, 65)	F	F	F	F	F	F	F	F	188	376
20. Watson (31, 31)	F	F	160	320	57	114	52	104	72	143
21. Extended Rosen. (4, 4)	69	138	164	328	69	138	65	130	131	262
22. Extended Powell sing. (4, 4)	F	F	79	158	39	78	35	70	43	86
23. Penalty I (6, 7)	F	F	448	896	69	139	123	245	273	546
24. Penalty II (5, 10)	21	42	94	188	19	38	16	32	19	38
25. Variably dimensioned (10, 12)	55	108	192	384	56	113	53	106	203	406
26. Trigonometric (10, 10)	F	F	56	112	F	F	F	F	F	F
27. Discrete bound. value (4, 4)	19	39	23	46	19	39	11	22	10	20
28. Discrete int. equation (20, 20)	14	29	26	52	8	17	8	17	12	24
29. Broyden tridiagonal (20, 20)	34	68	109	218	31	62	32	64	53	106
30. Broyden banded (10, 10)	42	83	119	238	38	76	31	62	56	112
31. Linear (10, 10)	1	2	1	2	1	2	1	2	1	2
32. Linear-1 (10, 10)	20	40	20	40	20	40	20	40	20	40
33. Linear-0 (10, 10)	19	19	18	36	18	36	18	36	18	36

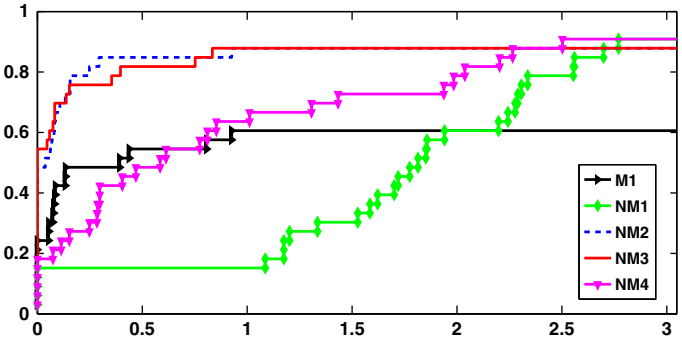


Fig. 1 Iteration performance profiles in a $\log_2$ scale

that strategies like (36) may be competitive in comparison with usual non-monotone approaches.

According to Theorems 1 and 4, the number of iterations NI_ϵ required by Algorithm 3 to find an ϵ-critical point of a possibly non-convex function f satisfies $NI_\epsilon \leq C\epsilon^{-2}$, where constant $C > 0$ depends on the problem and on the parameters in the algorithm. By assuming that

$$NI_\epsilon = C\epsilon^{-p},$$

we can estimate the complexity power p by using a simple formula similar to those used to determine the computational rate of convergence of numerical schemes for differential equations. In fact, fixed $h > 1$, by applying a given code to an objective function f, we can compute NI_ϵ and $NI_{\epsilon/h}$. Note that

$$\frac{NI_{\epsilon/h}}{NI_\epsilon} = \frac{C(\epsilon/h)^{-p}}{C\epsilon^{-p}} = h^p.$$

Thus, if $NI_{\epsilon/h} > NI_\epsilon$, we have

$$p = \frac{1}{\log(h)} \log\left(\frac{NI_{\epsilon/h}}{NI_\epsilon}\right). \tag{39}$$

Considering (39), we estimated the complexity power p taking as reference the monotone code M1 and the non-monotone code NM3. For that, we computed NI_ϵ and $NI_{\epsilon/h}$ with $\epsilon = 10^{-1}$ and $h = 10^4$. We allowed a maximum number of 5×10^5 iterations for both codes. The results for problems from [21] are given in Table 2. Note that, the estimated power p in all problems is smaller than 2 for both codes (the biggest power found was $p = 1.0644$). This is in accordance with Theorem 4, and also illustrate the very pessimistic aspect of these complexity bounds when we consider non-convex problems.

Table 2 Numerical estimation of the complexity order p in $NI_\epsilon = C\epsilon^{-p}$

PROBLEM	Monotone (M1)			Non-monotone (NM3)		
	NI_ϵ	$NI_{\epsilon/h}$	p	NI_ϵ	$NI_{\epsilon/h}$	p
1. Rosenbrock	38	45	0.0184	35	41	0.0172
2. Freudenstein-Roth	21	27	0.0273	23	26	0.0133
3. Powell badly scaled	F	F	–	157	178	0.0136
4. Brown badly scaled	77	F	–	96	96	–
5. Beale	14	17	0.0211	12	18	0.0440
6. Jennrich–Sampson	34	36	0.0062	44	46	0.0048
7. Helical valley	27	34	0.0250	31	35	0.0100
8. Bard	11	20	0.0649	11	20	0.0649
9. Gaussian	1	14	0.1505	1	4	0.1505
10. Meyer	F	F	–	F	F	–
11. Gulf	3	9	0.1193	3	10	0.1307
12. Box	8	30021	0.8936	10	30	0.1193
13. Powell singular	19	2100	0.5109	19	35	0.0663
14. Wood	20	90	0.1633	27	90	0.1307
15. Kowalik–Osborne	1	2314	0.8411	1	23	0.3404
16. Brown-Dennis	34	F	–	46	50	0.0091
17. Osborne 1	F	F	–	F	F	–
18. Biggs EXP6	6	38	0.2004	6	38	0.2004
19. Osborne 2	F	F	–	F	F	–
20. Watson	34	8491	0.5994	35	52	0.0430
21. Extended Rosenbrock	61	69	0.0134	62	65	0.0051
22. Extended Powell singular	19	2100	0.5109	19	35	0.0663
23. Penalty I	25	452530	1.0644	17	123	0.2149
24. Penalty II	9	21	0.0920	9	16	0.0625
25. Variably dimensioned	53	55	0.0040	50	53	0.0063
26. Trigonometric	F	F	–	F	F	–
27. Discrete boundary value	2	19	0.2444	2	11	0.1851
28. Discrete integral equation	2	14	0.2113	2	8	0.1505
29. Broyden tridiagonal	28	34	0.0211	28	32	0.0145
30. Broyden banded	29	42	0.0402	20	31	0.0476
31. Linear	1	1	–	1	1	–
32. Linear-1	20	20	–	20	20	–
33. Linear-0	19	19	–	18	18	–

Finally, in order to test the r-linear rate of convergence predicted by Theorem 6, we applied all non-monotone codes to a linear least squares problem

$$\min_{x\in\mathbb{R}^n} f(x) \equiv \frac{1}{2}\|Ax - b\|_2^2, \tag{40}$$

Table 3 Number of iterations NI_ϵ required to ensure $f(x_k) - f(x^*) \leq \epsilon$

ϵ	NM1	NM2	NM3	NM4
10^{-1}	363	87	110	558
10^{-2}	383	93	122	563
10^{-3}	402	100	129	570
10^{-4}	426	102	130	578
10^{-5}	444	108	137	585
10^{-6}	466	115	144	592
10^{-7}	496	121	150	599
10^{-8}	516	128	156	609
10^{-9}	525	139	163	627
10^{-10}	552	141	174	634

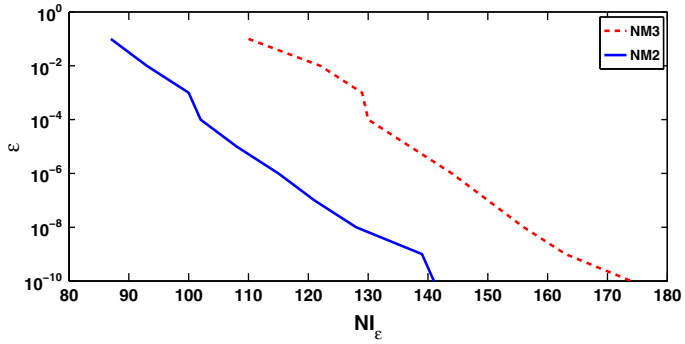


Fig. 2 Linear convergence behavior of the sequences of function values generated by codes NM2 and NM3

where $A \in \mathbb{R}^{m\times n}$ was a full rank matrix randomly generated, with $m = 5000$ and $n = 500$. To force $x^* = (1, \ldots, 1) \in \mathbb{R}^n$ to be the solution of (40), we defined $b = Ax^*$. Thus, we obtained $f(x^*) = 0$ as the minimum value of f. We also considered a randomly generated initial point. In this experiment, we use NI_ϵ to denote the number of iterations required by the code to ensure $f(x_k) - f(x^*) \leq \epsilon$. Table 3 present these numbers obtained for each code and for several values of ϵ. The results confirm the rate of convergent established in Theorem 6 for the algorithms corresponding to codes NM1, NM2 and NM3. The r-linear convergence becomes more evident in Fig. 2, where we plot the graph NI_ϵ *vs* ϵ for NM2 and NM3.

6 Conclusion

In this paper we investigate the worst-case complexity of the class of non-monotone line search algorithm proposed by Sachs and Sachs [24] for smooth unconstrained optimization. For a potentially non-convex objective function f, we proved that the general non-monotone algorithm can take at most $\mathcal{O}(\epsilon^{-2})$ iterations to ensure $\|\nabla f(x_k)\| \leq \epsilon$.

Assuming that the objective function is convex and that the steps in the sequence generated by the general algorithm are taken along the negative gradient direction, we proved a complexity bound of $\mathcal{O}(\epsilon^{-1})$ iterations for achieving $f(x_k) - f(x^*) \leq \epsilon$, where $f(x^*)$ is the global minimum of f. Furthermore, assuming that the objective function is gradient-dominated of degree 2 (e.g., strongly convex) and restricting our attention to a subclass of non-monotone algorithms with an upper bound α_{max} on α_k (including the one proposed by Zhang and Hager [27]), we proved a complexity bound of $\mathcal{O}(\log(\epsilon^{-1}))$ iterations to ensure $f(x_k) - f(x^*) \leq \epsilon$. These complexity bounds agree in order with those already proved for monotone first-order algorithms. Finally, we proposed a new way to dynamically adjust the monotonic and non-monotonic behavior of the sequence of function values, which is also covered by our complexity theory. Limited numerical results suggest that this new strategy may be competitive in comparison with the usual non-monotone approaches.

References

1. Ahookhosh, M., Amini, K., Bohrami, S.: A class of nonmonotone Armijo-type line search method for unconstrained optimization. Optimization **61**, 387–404 (2012)
2. Birgin, E.G., Martínez, J.M., Raydan, M.: Spectral projected gradient methods: review and pespectives. J. Stat. Softw. **60**, 1–21 (2014)
3. Birgin, E.G., Gardenghi, J.L., Martínez, J.M., Santos, S.A., Toint, P.L.: Worst-case evaluation complexity for unconstrained nonlinear optimization using high-order regularized models. Math. Program. **163**, 359–368 (2017)
4. Birgin, E.G., Martínez, J.M.: The use of quadratic regularization with a cubic descent condition for unconstrained optimization. SIAM J. Optim. **27**, 1049–1074 (2017)
5. Cartis, C., Gould, N.I.M., Toint, P.L.: Adaptive cubic regularization methods for unconstrained optimization. Part II: worst-case function—and derivative—evaluation complexity. Math. Program. **130**, 295–319 (2011)
6. Cartis, C., Gould, N.I.M., Toint, P.L.: Evaluation complexity of adaptive cubic regularization methods for convex unconstrained optimization. Optim. Methods Softw. **27**, 197–219 (2012)
7. Cartis, C., Sampaio, P.R., Toint, P.L.: Worst-case evaluation complexity of first-order non-monotone gradient-related algorithms for unconstrained optimization. Optimization **64**, 1349–1361 (2015)
8. Curtis, F.E., Robinson, D.P., Samadi, M.: A trust region algorithm with a worst-case iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$ for nonconvex optimization. Math. Program. doi:10.1007/s10107-016-1026-2 (2017)
9. Dodangeh, M., Vicente, L.N.: Worst case complexity of direct search under convexity. Math. Program. **155**, 307–332 (2016)
10. Dolan, E., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201–2013 (2002)
11. Dussault, J.-P.: ARCq: a new adaptive regularization by cubics variant. Optim. Methods Softw. doi:10.1080/10556788.2017.1322080 (2017)
12. Grapiglia, G.N., Nesterov, Y.: Regularized Newton methods for minimizing functions with Hölder continuous hessians. SIAM J. Optim. **27**, 478–506 (2017)
13. Grapiglia, G.N., Yuan, J., Yuan, Y.: On the convergence and worst-case complexity of trust-region and regularization methods for unconstrained optimization. Math. Program. **152**, 491–520 (2015)
14. Grapiglia, G.N., Yuan, J., Yuan, Y.: On the worst-case complexity of nonlinear stepsize control algorithms for convex unconstrained optimization. Optim. Methods Softw. **31**, 591–604 (2016)
15. Grapiglia, G.N., Yuan, J., Yuan, Y.: Nonlinear stepsize control algorithms: complexity bounds for first-and second-order optimality. J. Optim. Theory Appl. **171**, 980–997 (2016)
16. Gratton, S., Sartenaer, A., Toint, P.L.: Recursive trust-region methods for multiscale nonlinear optimization. SIAM J. Optim. **19**, 414–444 (2008)
17. Grippo, L., Lampariello, F., Lucidi, S.: A nonmonotone line search technique for Newton's method. SIAM J. Numer. Anal. **23**, 707–716 (1986)

18. Gu, N., Mo, J.: Incorporating nonmonotone strategies into the trust region method for unconstrained optimization. Comput. Math. Appl. **55**, 2158–2172 (2008)
19. Martínez, J.M., Raydan, M.: Cubic-regularization counterpart of a variable-norm trust-region method for unconstrained minimization. J. Global Optim. **68**, 367–385 (2017)
20. Mo, J., Liu, C., Yan, S.: A nonmonotone trust-region method based on nonincreasing technique of weighted average of the sucessive function value. J. Comput. Appl. Math. **209**, 97–108 (2007)
21. Moré, J.J., Garbow, B.S., Hillstrom, K.E.: Testing unconstrained optimization software. ACM Trans. Math. Softw. **7**, 17–41 (1981)
22. Nesterov, Y.: Introductory Lectures on Convex Optimization: A Basic Course. Kluwer Academic Publishers, Dordrecht (2004)
23. Nesterov, Y., Polyak, B.T.: Cubic regularization of Newton method and its global performance. Math. Program. **108**, 177–205 (2006)
24. Sachs, E.W., Sachs, S.M.: Nonmonotone line searches for optimization algorithms. Control Cybern. **40**, 1059–1075 (2011)
25. Sun, W., Yuan, Y.: Optimization theory and methods: nonlinear programming. Springer Optimization and its Application, vol. 1. Springer, New York (2006)
26. Vicente, L.N.: Worst case complexity of direct search. EURO J. Comput. Optim. **1**, 143–153 (2013)
27. Zhang, H.C., Hager, W.W.: A nonmonotone line search technique for unconstrained optimization. SIAM J. Optim. **14**, 1043–1056 (2004)