EURO Journal on Computational Optimization 10 (2022) 100044

Contents lists available at ScienceDirect

EURO Journal on Computational Optimization

journal homepage: www.elsevier.com/locate/ejco

A nonlinear conjugate gradient method with complexity guarantees and its application to nonconvex regression

Rémi Chan–Renous-Legoubin [a], Clément W. Royer [b,*,1]

[a] Université Paris Dauphine-PSL, 75016 Paris, France
[b] LAMSADE, CNRS, Université Paris Dauphine-PSL, 75016 Paris, France

ARTICLE INFO

ABSTRACT

Nonlinear conjugate gradients are among the most popular techniques for solving continuous optimization problems. Although these schemes have long been studied from a global convergence standpoint, their worst-case complexity properties have yet to be fully understood, especially in the nonconvex setting. In particular, it is unclear whether nonlinear conjugate gradient methods possess better guarantees than first-order methods such as gradient descent. Meanwhile, recent experiments have shown impressive performance of standard nonlinear conjugate gradient techniques on certain nonconvex problems, even when compared with methods endowed with the best known complexity guarantees.

In this paper, we propose a nonlinear conjugate gradient scheme based on a simple line-search paradigm and a modified restart condition. These two ingredients allow for monitoring the properties of the search directions, which is instrumental in obtaining complexity guarantees. Our complexity results illustrate the possible discrepancy between nonlinear conjugate gradient methods and classical gradient descent. A numerical investigation on nonconvex robust regression problems as well

* Corresponding author.
E-mail addresses: remi.chan-renous-legoubin@dauphine.eu (R. Chan–Renous-Legoubin), clement.royer@lamsade.dauphine.fr (C.W. Royer).

[1] Support for this research was provided by CNRS INS2I under the grant GASCON and by Agence Nationale de la Recherche through program ANR-19-P3IA-0001 (PRAIRIE 3IA Institute).

https://doi.org/10.1016/j.ejco.2022.100044

as a standard benchmark illustrate that the restarting condition can track the behavior of a standard implementation.

1. Introduction

In this paper, we are interested in the problem

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

where $f:\mathbb{R}^n\to\mathbb{R}$ is a Lipschitz continuously differentiable, nonconvex function. Given a tolerance $\epsilon\in(0,1)$, our goal is to compute an ϵ-approximate stationary point, that is, a vector $x\in\mathbb{R}^n$ such that

$$\|\nabla f(x)\|\leq\epsilon, \tag{2}$$

where $\|\cdot\|$ denotes the Euclidean norm in $\mathbb{R}^n$. We are interested in algorithms that provably reach a point satisfying (2) in a finite amount of computational work (number of iterations, and evaluations of f and its derivatives). Such results are termed *worst-case complexity guarantees*, and have been a growing area of interest in nonconvex optimization, fueled by the interest of such results in data analysis and statistics [36].

The most classical algorithm for solving problem (1) is gradient descent, that proceeds by moving along the negative gradient direction. It is known that such a method typically reaches a point satisfying (2) in at most $\mathcal{O}(\epsilon^{-2})$ iterations or gradient evaluations.[2] This bound contrasts with its counterparts in convex and strongly convex problems, and is sharp for gradient descent, even when exact line searches are used [7,11]. Perhaps more surprisingly, the bound $\mathcal{O}(\epsilon^{-2})$ is also sharp for more elaborate frameworks, such as trust region and Newton's method [7]. Under additional regularity assumptions on the objective function, it is possible to design algorithms with better properties, that typically require higher-order information to be used explicitly in the algorithm. Cubic regularization methods, that achieve an iteration complexity in $\mathcal{O}(\epsilon^{-3/2})$, were the first class of algorithms to be equipped with such a guarantee in the nonconvex setting [30,8]. These results attracted quite a lot of attention, leading to multiple second-order methods with similar complexity bounds being proposed [3,9,10,13,14,34,35]. Although out of the scope of this work, we note that the complexity results can be further improved by leveraging high-order smoothness [2].

[2] Throughout this paper, we write $\mathcal{O}(A)$ to indicate a constant times A, where the terms in the constants do not depend on that in A. The notation $\tilde{\mathcal{O}}(A)$ stands for $\mathcal{O}(\log^c(A)A)$ with $c>0$.

More recently, accelerated gradient techniques combined with negative curvature estimation procedures, for which a gradient evaluation complexity in $\tilde{\mathcal{O}}(\epsilon^{-7/4})$ can be shown [4,5]. The latter methods depart from standard nonlinear optimization techniques, and were shown to be outperformed by a standard nonlinear conjugate gradient implementation on a nonconvex regression example [4]. However, their construction facilitates the derivation of complexity results. Providing a complexity analysis of standard nonlinear optimization schemes, on the other hand, remains a challenging endeavor.

Conjugate gradient (CG) methods are an example of popular nonlinear optimization techniques that have yet to be endowed with a comprehensive complexity analysis. When applied to strongly convex quadratics, those methods typically reduce to linear conjugate gradient, and complexity guarantees have long been known in that setting [31]. Recent results have shown that linear conjugate gradient can also be used in conjunction with a Newton-type framework when the objective is Lipschitz twice continuously differentiable. The resulting methods can be analyzed from a complexity viewpoint on nonconvex optimization problems, leading to complexity bounds that match the best known in the literature for this class of problems [12,33,34].

The situation becomes quite different when considering nonlinear conjugate gradient techniques. Numerous conditions for global convergence of nonlinear CG schemes have been proposed in the literature [15,19–23], yet early complexity analyzes showed that nonlinear CG techniques could have worse guarantees than gradient descent on strongly convex problems [29]. Still, recent proposals have combined modern nonlinear conjugate gradient with accelerated gradient tools to yield a method that both reduces to linear CG on quadratics and converges as fast as accelerated gradient on convex and strongly convex problems [24,25,27]. Nevertheless, deriving complexity guarantees for classical nonlinear CG variants remains a difficult task, particularly in the nonconvex setting.

In this paper, we propose a nonlinear conjugate gradient framework for optimizing nonconvex functions with complexity guarantees. By adapting the classical restart condition of nonlinear conjugate gradient methods, we are able to provide decrease guarantees at every iteration, that depend on whether the restart condition is triggered. Overall, our method is shown to depart from gradient descent on certain iterations, for which our analysis provides better guarantees. To investigate the practical impact of this condition, we first confirm earlier findings about the performance of these methods on nonconvex regression tasks, where regimes in which the restart condition is representative of the algorithmic behavior of nonlinear conjugate gradient can be identified. We also investigate the difference between classical implementations of nonlinear CG that lack complexity guarantees and our framework on a nonlinear optimization benchmark.

The rest of this paper is organized as follows. In Section 2, we recall the key features of nonlinear conjugate gradient algorithms, and we describe our framework based on a modified restart condition. Complexity results for this framework are obtained and discussed in Section 3. In Section 4, we then conduct a numerical study of our proposed method involving both nonconvex regression tasks and comparisons on a nonlinear optimization test set. Final comments are made in Section 5.

2. Nonlinear conjugate gradient framework

In this section, we describe a nonlinear conjugate gradient method based on Armijo line-search and a modified restart condition. To this end, we first recall the main features of nonlinear conjugate gradient methods, then provide a description of our proposed scheme.

2.1. Nonlinear conjugate gradient

Nonlinear conjugate gradient techniques [23,32] are iterative optimization schemes of the form

$$x_{k+1} = x_k + \alpha_k d_k, \tag{3}$$

where the direction d_k is a search direction, and α_k is a stepsize typically computed through a line search. Several line-search strategies have been proposed in the literature [15,23]. In this paper, we focus on using a backtracking Armijo line search, since this line search led to good performance on nonconvex regression problems [4].

The goal of a conjugate gradient approach is to combine local information (i.e. the negative gradient at the current point) with the *previous direction*, that is possibly still relevant for the next iterate. As a result, a nonlinear CG method selects $d_0 = -\nabla f(x_0)$ and

$$\begin{cases} d_0 = -\nabla f(x_0) \\ d_{k+1} = -\nabla f(x_{k+1}) + \beta_{k+1} d_k \quad \forall k \in \mathbb{N}. \end{cases} \tag{4}$$

The choice of the formula for the parameter β_{k+1} gives rise to various nonlinear CG schemes [22]. The most popular choices for β_{k+1} include the *Fletcher-Reeves* formula

$$\beta_{k+1}^{FR} := \frac{\|\nabla f(x_{k+1})\|^2}{\|\nabla f(x_k)\|^2} \tag{5}$$

and the *Polak-Ribière* (also known as Polak-Ribière-Polyak) formula

$$\beta_{k+1}^{PR} := \frac{\nabla f(x_{k+1})^T (\nabla f(x_{k+1}) - \nabla f(x_k))}{\|\nabla f(x_k)\|^2}. \tag{6}$$

A popular variant of the Polak-Ribière formula is the *PRP+* update:

$$\beta_{k+1}^{PRP+} := \max\left\{ \frac{\nabla f(x_{k+1})^T (\nabla f(x_{k+1}) - \nabla f(x_k))}{\|\nabla f(x_k)\|^2}, 0 \right\}. \tag{7}$$

Using the PRP+ formula has been shown to guarantee global convergence of a nonlinear conjugate gradient method with appropriate line-search condition [17]. More recently, the formula proposed by Hager and Zhang [22]

$$\beta_{k+1}^{HZ} := \frac{1}{d_k^{\mathrm{T}} y_k} \left(y_k - 2 d_k \frac{\|y_k\|^2}{d_k^{\mathrm{T}} y_k} \right)^{\mathrm{T}} \nabla f(x_{k+1}), \tag{8}$$

where $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$, has been found quite successful in practice.

Regardless of the variant that is used, a nonlinear conjugate gradient method can produce iterates such that $\nabla f(x_{k+1})^T d_{k+1} \geq 0$, in which case there is no guarantee for decrease in the direction d_{k+1}: a typical fix consists in redefining the search direction as $d_{k+1} = -\nabla f(x_{k+1})$. This process is called restarting, and is a common feature of a nonlinear conjugate gradient implementation. In the next section, we propose an alternative to this restart condition that endows a conjugate gradient scheme with complexity results.

2.2. Nonlinear CG with modified restart

In our framework, we build on the restarting idea by monitoring the value of $\nabla f(x_k)^{\mathrm{T}} d_k$ and that of $\|d_k\|$. Algorithm 1 describes our framework. At every iteration, we perform a backtracking line search to compute a step that yields a suitable decrease in the objective function (see condition (9)). Once the new point has been computed, we evaluate the gradient at the next iterate, as well as the parameter β_{k+1}, that is typically chosen from one of the formulas given in the previous section. Both the gradient and the parameter are then used to define the new search direction.

Algorithm 1 Nonlinear conjugate gradient with modified restart condition.

Inputs: $x_0 \in \mathbb{R}^d$, $\eta \in (0,1)$, $\theta \in (0,1)$, $\sigma \in (0,1]$, $\kappa \geq 1$, $p \geq 0$, $q \geq 0$.
Set $g_0 = \nabla f(x_0)$, $d_0 = -g_0$, $k = 0$.
for $k = 0, 1, 2, \ldots$ **do**
Compute $\alpha_k = \theta^{j_k}$ where j_k is the smallest nonnegative integer such that

$$f(x_k + \alpha_k d_k) < f(x_k) + \eta \alpha_k g_k^T d_k. \tag{9}$$

Set $x_{k+1} = x_k + \alpha_k d_k$ and $g_{k+1} = \nabla f(x_{k+1})$.
Choose a conjugate direction parameter β_{k+1}.
Set $d_{k+1} = -g_{k+1} + \beta_{k+1} d_k$.
If the condition

$$g_{k+1}^{\mathrm{T}} d_{k+1} \geq -\sigma \|g_{k+1}\|^{1+p} \quad \text{or} \quad \|d_{k+1}\| \geq \kappa \|g_{k+1}\|^q, \tag{10}$$

holds, restart the algorithm by setting $d_{k+1} = -g_{k+1}$.
end for

The key ingredient to Algorithm 1 is the restarting condition (10), that determines whether the nonlinear CG direction is kept for the next iteration. For any $k \geq 0$, if iteration k does not end with a restart, we have[3]

[3] Although the inequalities should be strict, we use non-strict inequalities for notational convenience.

$$g_{k+1}^{\mathrm{T}} d_{k+1} \leq -\sigma \|g_{k+1}\|^{1+p} \quad \text{and} \quad \|d_{k+1}\| \leq \kappa \|g_{k+1}\|^{q}. \tag{11}$$

In Section 3, we will establish that these inequalities together with the line search allow for proving complexity bounds for Algorithm 1. Note that the above properties can be viewed as a more general case of the following condition (obtained for $p = q = 1$):

$$g_{k+1}^{\mathrm{T}} d_{k+1} \leq -\sigma \|g_{k+1}\|^{2} \quad \text{and} \quad \|d_{k+1}\| \leq \kappa \|g_{k+1}\|.$$

Such a condition is typical of gradient-related directions, and has been instrumental in obtaining complexity guarantees for gradient-type methods [11]. In the context of nonlinear conjugate gradient method, similar properties have been used to establish global convergence [20]. A case of particular interest to us is $d_k = -g_k$, which occurs when the restarting process is triggered.

3. Complexity analysis

In this section, we derive a complexity result for our restarted variant of nonlinear conjugate gradient. Section 3.1 provides the necessary assumptions as well as intermediate results, while Section 3.2 establishes and discusses complexity bounds for our algorithm.

3.1. Decrease lemmas

We make the following assumptions about the objective function of problem (1).

Assumption 3.1. The function f is continuously differentiable on $\mathbb{R}^n$ and its gradient is L-Lipschitz continuous for $L > 0$.

Assumption 3.2. There exists $f_{\mathrm{low}} \in \mathbb{R}$ such that $f(x) \geq f_{\mathrm{low}}$ for every $x \in \mathbb{R}^n$.

Our complexity analysis relies on partitioning the iterations into $\mathcal{R} \cup \mathcal{N}$, where

$$\begin{array}{rcl} \mathcal{N} & = & \{k \in \mathbb{N} \mid g_k^{\mathrm{T}} d_k \leq -\sigma \|g_k\|^{1+p} \quad \text{and} \quad \|d_k\| \leq \kappa \|g_k\|^{q}\} \\ \mathcal{R} & = & \mathbb{N} \setminus \mathcal{N}. \end{array} \tag{12}$$

For $k \geq 1$, the fact that $k \in \mathcal{N}$ is checked explicitly within our algorithm. If $k \in \mathcal{R}$, the restarting process is triggered and the search direction will be the negative gradient. For this reason, we say that $k \in \mathcal{R}$ is the index of a *restarted iteration*, while $k \in \mathcal{N}$ is the index of a *non-restarted iteration*. Depending on the nature of each iteration, we can bound the number of backtracking steps needed to compute a suitable stepsize. We begin by the non-restarted iterations, as the proof encompasses that of restarted iterations.

Lemma 3.1. *Let Assumption 3.1 hold, and let $k \in \mathcal{N}$ such that $\|g_k\| > 0$. Then, the line-search process terminates after at most $\lfloor \bar{j}_{\mathcal{N},k} + 1 \rfloor$ iterations, where*

$$\bar{j}_{\mathcal{N},k} := \left[\log_\theta \left(\frac{2(1-\eta)\sigma}{\kappa^2 L} \right) \|g_k\|^{1+p-2q} \right]_+ . \tag{13}$$

Moreover, the resulting decrease at the kth iteration satisfies

$$f(x_k) - f(x_{k+1}) > c_{\mathcal{N}} \min \left\{ \|g_k\|^{1+p}, \|g_k\|^{2(1+p-q)} \right\}, \tag{14}$$

where

$$c_{\mathcal{N}} := \eta\sigma \min \left\{ 1, \frac{2(1-\eta)\sigma\theta}{\kappa^2 L} \right\}.$$

Proof. If the decrease condition (9) holds for $\alpha_k = 1$, then the bound (13) holds. Moreover, combining (9) with the definition of $\mathcal{N}$ in (12) gives

$$f(x_k) - f(x_{k+1}) > -\eta\alpha_k g_k^{\mathrm{T}} d_k \geq \eta\sigma \|g_k\|^{1+p}, \tag{15}$$

hence (14) also holds.

Suppose now that the line-search condition (9) fails for some $\alpha = \theta^j$ with $j \in \mathbb{N}$. Using a Taylor expansion of f at x_k (see, e.g. [36, (4.1.2)]), we have

$$f(x_k + \alpha d_k) \leq f(x_k) + \alpha g_k^{\mathrm{T}} d_k + \frac{L}{2}\alpha^2 \|d_k\|^2.$$

Thus,

$$\begin{aligned} \eta\alpha g_k^{\mathrm{T}} d_k &\leq f(x_k + \alpha d_k) - f(x_k) \\ &\leq \alpha g_k^{\mathrm{T}} d_k + \frac{L}{2}\alpha^2 \|d_k\|^2 \\ &\leq \alpha g_k^{\mathrm{T}} d_k + \frac{\kappa^2 L}{2}\alpha^2 \|g_k\|^{2q}, \end{aligned}$$

where the last inequality comes from (12). Re-arranging the terms, we obtain

$$\begin{aligned} -(1-\eta)\alpha g_k^{\mathrm{T}} d_k &\leq \frac{\kappa^2 L}{2}\alpha^2 \|g_k\|^{2q} \\ (1-\eta)\alpha\sigma \|g_k\|^{1+p} &\leq \frac{\kappa^2 L}{2}\alpha^2 \|g_k\|^{2q} \\ \frac{2(1-\eta)\sigma}{\kappa^2 L} \|g_k\|^{1+p-2q} &\leq \alpha. \end{aligned} \tag{16}$$

The condition (16) can only hold for $j \leq \bar{j}_{\mathcal{N},k}$: as a result, the line-search process must terminate after $j_k \leq \lfloor \bar{j}_{\mathcal{N},k} + 1 \rfloor$ iterations. Moreover, since the line search did not terminate after $j_k - 1$ iterations, we have

$$\theta^{j_k-1} \geq \frac{2(1-\eta)\sigma}{\kappa^2 L}\|g_k\|^{1+p-2q} \quad \Leftrightarrow \quad \alpha_k = \theta^{j_k} \geq \frac{2(1-\eta)\theta\sigma}{\kappa^2 L}\|g_k\|^{1+p-2q}.$$

Consequently, the function decrease at iteration k satisfies:

$$f(x_k) - f(x_{k+1}) > -\eta\alpha_k g_k^{\mathrm{T}} d_k \geq \frac{2\eta(1-\eta)\theta\sigma^2}{\kappa^2 L}\|g_k\|^{2(1+p-q)} \geq c_{\mathcal{N}}\|g_k\|^{2(1+p-q)}, \quad (17)$$

hence (14) also holds in this case. □

We now consider the restarted iterations. In that case, we have $d_k = -g_k$, implying that

$$g_k^{\mathrm{T}} d_k = -\|g_k\|^2 \quad \text{and} \quad \|d_k\| = \|g_k\|. \quad (18)$$

Thus, for the restarted iterations, the search direction satisfies a property analogous to that defining non-restarted iterations in (12) with $\sigma = \kappa = 1$ and $p = q = 1$. A reasoning identical to that used in the proof of Lemma 3.1 leads to the following result.

Lemma 3.2. *Let Assumption 3.1 hold, and let $k \in \mathcal{R}$ such that $\|g_k\| > 0$. Then, the line-search process terminates after at most $\bar{j}_{\mathcal{R}} + 1$ iterations, where*

$$\bar{j}_{\mathcal{R}} := \left\lfloor \log_\theta \left(\frac{2(1-\eta)}{L} \right) \right\rfloor_+ . \quad (19)$$

Moreover, the resulting decrease at the kth iteration satisfies

$$f(x_k) - f(x_{k+1}) > c_{\mathcal{R}}\|g_k\|^2, \quad (20)$$

where

$$c_{\mathcal{R}} := \eta \min\left\{1, \frac{2(1-\eta)\theta}{L}\right\}.$$

The result of Lemmas 3.1 and 3.2 are instrumental to bounding the number of iterations necessary to reach an approximate stationary point.

3.2. Main results

Our main result is a bound on the number of iterations performed by the algorithm prior to reaching an ϵ-stationary point: this bound also applies to the number of gradient evaluations.

Theorem 3.1. *Let Assumptions 3.1 and 3.2 hold. Suppose that $1 + p - q \geq 0$. Then, the number of iterations (and objective gradient evaluations) required by Algorithm 1 to reach a point satisfying (2) is at most*

$$K_\epsilon := \left\lfloor \frac{f(x_0) - f_{\text{low}}}{c_{\mathcal{R}}} \epsilon^{-2} + \frac{f(x_0) - f_{\text{low}}}{c_{\mathcal{N}}} \epsilon^{-\max\{1+p, 2(1+p-q)\}} \right\rfloor. \tag{21}$$

Proof. Let $K \in N$ be such that $\|\nabla f(x_k)\| > \epsilon$ for any $k = 0, \dots, K-1$. Following our partitioning (12), we define the index sets

$$\mathcal{N}_K := \mathcal{N} \cap \{0, \dots, K-1\},$$
$$\mathcal{R}_K := \mathcal{R} \cap \{0, \dots, K-1\}.$$

For any $k \in \mathcal{N}_K$, the result of Lemma 3.1 applies, and we have

$$f(x_k) - f(x_{k+1}) \geq c_{\mathcal{N}} \min\left\{ \|g_k\|^{1+p}, \|g_k\|^{2(1+p-q)} \right\} \geq c_{\mathcal{N}} \epsilon^{\max\{1+p, 2(1+p-q)\}}. \tag{22}$$

On the other hand, if $k \in \mathcal{R}_K$, applying Lemma 3.2 gives

$$f(x_k) - f(x_{k+1}) \geq c_{\mathcal{R}} \|g_k\|^2 \geq c_{\mathcal{R}} \epsilon^2. \tag{23}$$

We now consider the sum of function changes over all $k \in \{0, \dots, K-1\}$. By Assumption 3.2, we obtain

$$\begin{aligned} f(x_0) - f_{\text{low}} &\geq f(x_0) - f(x_K) \\ &\geq \sum_{k=0}^{K-1} [f(x_k) - f(x_{k+1})] \\ &\geq \sum_{k \in \mathcal{N}_K} [f(x_k) - f(x_{k+1})] + \sum_{k \in \mathcal{R}_K} [f(x_k) - f(x_{k+1})] \\ &> \sum_{k \in \mathcal{N}_K} c_{\mathcal{N}} \epsilon^{\max\{1+p, 2(1+p-q)\}} + \sum_{k \in \mathcal{R}_K} c_{\mathcal{R}} \epsilon^2. \end{aligned}$$

Since the right-hand side consists in two sums of positive terms, the above inequality implies that

$$f(x_0) - f_{\text{low}} > \sum_{k \in \mathcal{N}_K} c_{\mathcal{N}} \epsilon^{\max\{1+p, 2(1+p-q)\}} \Leftrightarrow |\mathcal{N}_K| < \frac{f(x_0) - f_{\text{low}}}{c_{\mathcal{N}}} \epsilon^{-\max\{1+p, 2(1+p-q)\}}$$

and

$$f(x_0) - f_{\text{low}} > \sum_{k \in \mathcal{R}_K} c_{\mathcal{R}} \epsilon^2 \Leftrightarrow |\mathcal{R}_K| < \frac{f(x_0) - f_{\text{low}}}{c_{\mathcal{R}}} \epsilon^{-2}.$$

Using $|\mathcal{N}_K| + |\mathcal{R}_K| = K$ finally yields

$$K < \frac{f(x_0) - f_{\text{low}}}{c_{\mathcal{R}}} \epsilon^{-2} + \frac{f(x_0) - f_{\text{low}}}{c_{\mathcal{N}}} \epsilon^{-\max\{1+p, 2(1+p-q)\}},$$

hence $K \leq K_\epsilon$. □

Note that the complexity bound of Theorem 3.1 also guarantees global convergence of the algorithmic framework, since it holds for ϵ arbitrarily close to 0. More precisely, it is possible to show that

$$\liminf_{k\to\infty} \|\nabla f(x_k)\| = 0,$$

which is a typical convergence result for nonlinear conjugate gradient using line search.

By combining the result of Theorem 3.1 with that of the Lemma 3.1 and 3.2, we can also provide an evaluation complexity bound of Algorithm 1.

Corollary 3.1. *Under the assumptions of Theorem 3.1, suppose further that* $1+p-2q = 0$. *Then, the number of function evaluations required by Algorithm 1 to reach a point satisfying* (2) *is at most*

$$\left\lfloor \left[\log_\theta \left(\frac{2(1-\eta)\sigma}{\kappa^2 L}\right)\right]_+ + 1 \right\rfloor K_\epsilon, \tag{24}$$

where K_ϵ *is defined in* (21).

Proof. Since $1+p-2q=0$, we have

$$\forall k \in \mathcal{N}, \quad \bar{j}_{\mathcal{N},k} = \left[\log_\theta \left(\frac{2(1-\eta)\sigma}{\kappa^2 L}\right)\right]_+,$$

hence this quantity is independent of the iteration index k. Moreover,

$$\max\left\{\left[\log_\theta \left(\frac{2(1-\eta)\sigma}{\kappa^2 L}\right)\right]_+, \bar{j}_{\mathcal{R}}\right\} = \left[\log_\theta \left(\frac{2(1-\eta)\sigma}{\kappa^2 L}\right)\right]_+$$

since $\kappa \geq 1$ and $\sigma \leq 1$. As a result, any iteration requires at most

$$\left\lfloor \left[\log_\theta \left(\frac{2(1-\eta)\sigma}{\kappa^2 L}\right)\right]_+ + 1 \right\rfloor$$

function evaluations. Combining this number with the result of Theorem 3.1 completes the proof. □

Note that the additional condition $1+p-2q=0$ can be replaced by a boundedness assumption on the gradient norm. Such an assumption also removes the need for the condition $1+p-q \geq 0$ in Theorem 3.1.

Table 1
Possible complexity values for different values of p and q satisfying $1 + p - 2q = 0$ and $1 + p - q \geq 0$.

p	1	3/4	1/2	1/4	0
$q = (1 + p)/2$	1	7/8	3/4	5/8	1/2
Order $\epsilon^{-(1+p)}$	ϵ^{-2}	$\epsilon^{-7/4}$	$\epsilon^{-3/2}$	$\epsilon^{-5/4}$	ϵ^{-1}

3.3. Interpretation of the complexity bounds

Both the iteration complexity bound and the evaluation complexity bound of Algorithm 1 are of order

$$\mathcal{O}\left(\epsilon^{-2}\right) + \mathcal{O}\left(\epsilon^{-\max\{1+p,2(1+p-q)\}}\right). \tag{25}$$

The proof of Theorem 3.1 shows that each term in the bound relates to a different form of iteration. The first part of the bound corresponds to negative gradient steps due to a restart in the algorithm. The other part corresponds to "true" conjugate gradient directions that satisfy condition (11): thanks to this condition, we can certify a different decrease formula for these iterations. As a result, our complexity analysis interpolates between that of gradient descent and that of an ideal method where no restart would occur. Note that the two terms in the maximum coincide when $1 + p - 2q = 0$: this condition also guarantees a constant number of backtracking line-search iterations.

Table 1 shows several possible choices for p and q that satisfy the requirements of our analysis. We focus on choices for which $1 + p \leq 2$, since those lead to a better complexity bound for the number of non-restarted iterations. Choosing p between 0 and 1 implies that the bound varies between ϵ^{-2} and ϵ^{-1}, suggesting that the overall number of iterations could be better than that of gradient descent. However, regardless of the choice of p, there is no a priori guarantee that the number of non-restarted iterations will be significantly larger than that of restarted iterations. In the next section, we investigate this behavior in the context of nonconvex regression problems.

4. Numerical illustration

In this section, we investigate the numerical behavior of our nonlinear conjugate gradient algorithm with the modified restarting condition. The purpose of these experiments is twofold. On one hand, we aim at better understanding the efficiency of nonlinear conjugate gradient on nonconvex regression problems, as demonstrated by Carmon et al. [4]. We revisit this experiment in Section 4.2. On the other hand, we compare our modified nonlinear conjugate gradient algorithm with standard variants on a benchmark of nonlinear optimization problems from the CUTEst collection [18], so as to assess the numerical impact of enforcing complexity guarantees. This is the purpose of Section 4.3. The setup for our experiments is described in Section 4.1.

4.1. Algorithms and implementation

Our experiments included the following methods:

- Semi-adaptive GD, a version of gradient descent using an adaptive estimate of the Lipschitz constant, that proceeds similarly to performing an Armijo linesearch [4].
- Armijo GD, a version of gradient descent using the Armijo line search procedure described in Algorithm 1.
- Standard NCG, a nonlinear conjugate gradient method with Armijo line search. In this variant, restart occurs whenever $g_{k+1}^{\mathrm{T}} d_{k+1} \geq 0$, in which case $d_{k+1} \leftarrow -g_{k+1}$. (Note that this corresponds to setting $\sigma = 0$ and $\kappa = \infty$ in Algorithm 1, but we single out this variant for simplicity.)
- Orthog NCG, a nonlinear conjugate gradient method with Armijo line search. Restarting occurs there whenever $|g_k^{\mathrm{T}} g_{k+1}| \geq \sigma \|g_k\|^2$ with $\sigma = 0.01$. This condition measures the loss of orthogonality between successive gradients [31, Chapter 5]. Although more elaborate variants based on this condition can be considered [26], we use one that merely restarts based on a single test, akin to our proposed method.
- Restarted NCG (p), our implementation of Algorithm 1 with $\sigma = 0.01$, $\kappa = 100$, $q = \frac{1+p}{2}$ and a variable value for p.

All variants of NCG were tested with the four formulas for β_{k+1} given in (5)–(8). The parameters of the Armijo line search were set as $\eta = \theta = 0.5$ for all methods. The line search was used with initial value 1 at iteration 0, then the value $2\alpha_k$ was used as initial value for iteration $k+1$. This procedure was not used for the Semi-adaptive GD method.

All the algorithms were implemented in MATLAB R2021a. Experiments were run on a Dell Latitude 7400 running Ubuntu 20.04 with Intel(R) Core(TM) i7-8665U CPU @ 1.90GHz and 31.2GB of memory.

4.2. Nonconvex regression tasks

Our first set of experiments follows the setup of Carmon et al. [4], in which a basic nonlinear conjugate gradient was found to be quite efficient on randomly generated nonconvex regression problems. Each problem instance corresponds to a dataset $\{(a_i, b_i)\}_{i=1}^m$, where $a_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}^m$, with $n = 30$ and $m = 60$. Every vector a_i is generated according to a Gaussian distribution of zero mean and identity covariance matrix, which we denote by $a_i \sim \mathcal{N}(0, I_n)$, where I_n is the identity matrix in $\mathbb{R}^{n\times n}$. Letting $A = [a_i^{\mathrm{T}}] \in \mathbb{R}^{m\times n}$ and $b = [b_i]_{i=1}^m$, we generate the vector b according to the following formula:

$$b = Az + 3\nu_1 + \nu_2,$$

where $z \sim \mathcal{N}(0, 4I_n)$, $\nu_1 \sim \mathcal{N}(0, I_m)$, and $\nu_2 \in \mathbb{R}^m$ has i.i.d. components drawn following a Bernoulli distribution of parameter 0.3.

Given a dataset $\{(a_i, b_i)\}_{i=1}^m$ of this form, we consider the robust regression problem:

$$\min_{x \in \mathbb{R}^d} f_{SB}(x) := \frac{1}{m} \sum_{i=1}^{m} \phi(a_i^{\mathrm{T}} x - b_i), \tag{26}$$

where $\phi : \mathbb{R} \to [0, \infty)$ is a smoothed biweight loss function defined by

$$\phi(t) = \frac{t^2}{1 + t^2}.$$

Carmon et al. [4] introduced function ϕ as a smooth proxy for the Tukey biweight loss function [1]. This terminology denotes a family of functions parameterized by $c > 0$ as follows:

$$\forall t \in \mathbb{R}, \qquad \rho_c(t) = \begin{cases} \frac{t^6}{6c^4} - \frac{t^4}{2c^2} + \frac{t^2}{2} & \text{if } |t| \leq c, \\ \frac{c^2}{6} & \text{otherwise.} \end{cases}$$

Given the same data $\{(a_i, b_i)\}_{i=1}^m$ than that used to define problem (26), we also consider the problem

$$\min_{x \in \mathbb{R}^d} f_{TB}(x) := \frac{1}{m} \sum_{i=1}^{m} \rho_{\sqrt{6}}(a_i^{\mathrm{T}} x - b_i), \tag{27}$$

where we chose $c = \sqrt{6}$ so as to be close to the smoothed version of the biweight loss. The solution of problem (27) belongs to the class of robust M-estimators in statistics; compared to a standard linear least-squares formulation, problem (27) is more robust to outliers in the data. Fig. 1 shows the shape of ϕ and $\rho_{\sqrt{6}}$. Both losses are nonconvex, resulting in both functions f_{SB} and f_{TB} being nonconvex. Those functions also satisfy Assumption 3.1. Note that both functions are twice continuously differentiable, but that only f_{SB} is infinitely smooth.

Results We generated 1000 datasets for linear regression according to the procedure above. For each dataset, we consider the associated nonconvex regression problems (26) and (27), and compare gradient descent techniques with standard NCG and several instances of restarted NCG (Algorithm 1). For every problem, all methods were run until $\|g_k\| \leq \epsilon = 10^{-4}$ or a budget of 10000 iterations was exhausted. We conducted experiments with values of p uniformly distributed between 0 and 1. In all cases, the performance was consistently better for $p \geq 0.5$, with best behavior typically obtained for $p \in [0.5, 0.75]$. We thus elected to present results using $p \in \{0, 0.25, 0.5, 0.75, 1\}$, as those values are representative of the behavior of our framework.

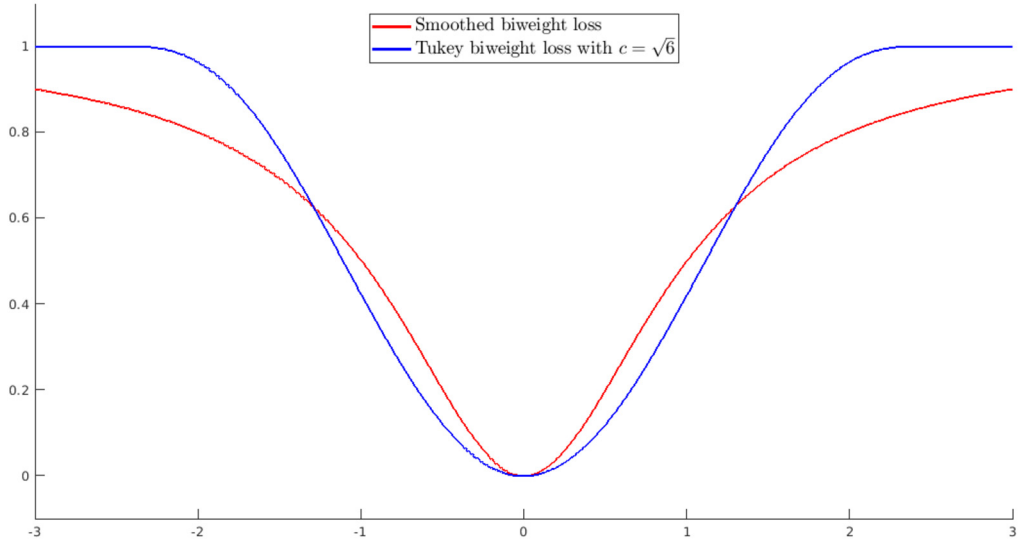

Fig. 1. Nonconvex losses for robust linear regression.

We begin by considering the original setup of Carmon et al. [4], where nonlinear CG was applied to problem (26) using the PRP+ formula (7). Table 2 shows the average percentage of restarted iterations among all iterations for each of the tested methods. The standard NCG method has a remarkably low percentage of restarted iterations, i.e. a very small fraction of iterations produced directions that were not of descent type. When we use the modified restarting condition of Algorithm 1 with $p < 0.5$, we observe that restarted iterations form the majority of iterations on average. This behavior suggests that the algorithm tends to use gradient descent directions, and that most directions produced by a nonlinear CG update satisfy the restart condition (10) when $p < 0.5$. On the contrary, when $p \geq 0.5$, the percentage of restarted iterations decreases significantly, indicating that the method relies on directions close to that of a standard nonlinear conjugate gradient method. Fig. 2 shows the fraction of instances solved as a function of the iteration budget, under the form of data profiles [28]. This figure confirms that the variants of Restarted NCG for $p \geq 0.5$ exhibit a behavior close to that of Standard NCG.

Fig. 3 illustrates the behavior of the minimum gradient norm on a representative run. The number and location of the restarted iterations (red circles) is the same for Standard NCG and the restarted NCG variants for $p \geq 0.5$, suggesting that the restarting condition (10) tends to be triggered for non-descent directions. On the contrary, using $p \in \{0, 0.25\}$ leads to series of restarted iterations, from which the algorithm does not appear to recover, in that the restarting condition keeps being triggered. These results strongly suggest that most directions produced by nonlinear CG do not satisfy condition (11) when p is small. However, we observe that in the early iterations, all methods perform

Table 2
Statistics on 1000 random instances of robust linear regression using smoothed biweight loss (problem (26)) and a budget of 10000 iterations. All variants use the PRP+ formula (7).

Method	Problems solved	Avg. restart (%)
Standard NCG	1000	0.74%
NCG(0)	1000	83.5%
NCG(0.25)	1000	53.2%
NCG(0.5)	1000	0.89%
NCG(0.75)	1000	0.76%
NCG(1)	1000	0.76%

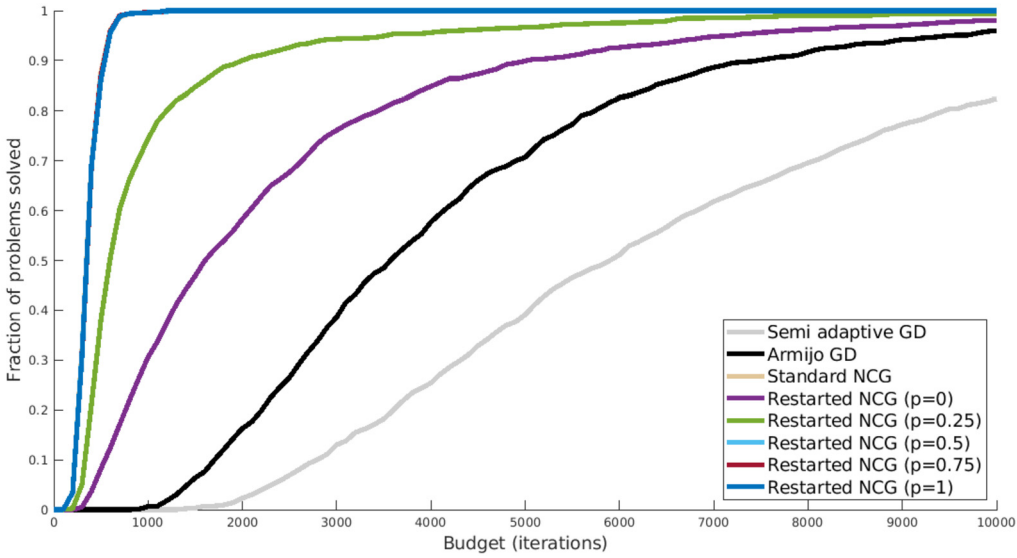

Fig. 2. Fraction of nonconvex problems with the smooth biweight loss (26) solved as function of the iteration budget. The curves corresponding to Restarted NCG with $p \in \{0.5, 0.75, 1\}$ overlap with that of Standard NCG. All NCG variants use the PRP+ formula (7).

non-restarted iterations, hence our conditions appear to be satisfied at the beginning of every run.

Table 3 as well as Figs. 4 and 5 are the counterparts of the previous results for the Tukey biweight loss problem (27). Although the percentages of restarted iterations are smaller than that of the previous table, the same overall trend can be observed, with a reduction in the number of restarted iterations as p increases, and a sharp decrease for $p \geq 0.5$. Fig. 5 also confirms that the restarted iterations for $p \geq 0.5$ occur at the same index as that of Standard NCG. We note that the discrepancy between nonlinear conjugate gradient methods and gradient descent methods is larger on problem (26), where the smoothed biweight function is used. Since the latter function is infinitely smooth, it is possible that high-order derivatives are relevant for optimization, in a way that nonlinear conjugate gradient better captures. We point out that the complexity analysis

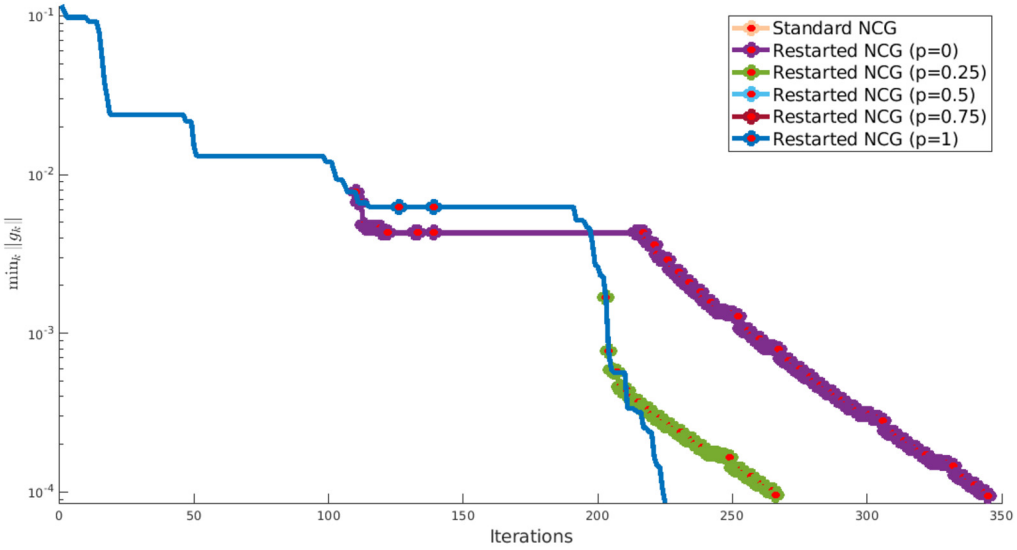

Fig. 3. Minimum gradient norm for a representative run using the smooth biweight loss (26) using the PRP+ formula (7). The curves corresponding to Restarted NCG with $p \in \{0.5, 0.75, 1\}$ overlap with that of Standard NCG, and all have the same 2 restarted iterations (blue circles with red filling). Restarted NCG($p = 0.25$) variant had 66 restarted iterations, while Restarted NCG($p = 0$) had 148 restarted iterations.

Table 3
Statistics on 1000 random instances of robust linear regression using Tukey loss (problem (27)) and a budget of 10000 iterations. All methods use the PRP+ formula (7).

Method	Problems solved	Avg. restart (%)
Standard NCG	1000	0.58%
NCG(0)	1000	62.7%
NCG(0.25)	1000	44.6%
NCG(0.5)	1000	3.47%
NCG(0.75)	1000	0.61%
NCG(1)	1000	0.63%

of Section 3 only assumes that the objective function is continuously differentiable, but that additional smoothness can improve guarantees of existing schemes [10,6].

Using the PR formula (6) yields very similar results to those for the PRP+ formula (7), therefore we do not report these results here. However, we provide results using the Hager-Zhang formula (8) in Tables 4 and 5. When this parameter formula is used, restart no longer occurs for Standard NCG as the direction is always a descent one [21]. Still, for the restarted NCG variants, we again observe that the average percentage of restarted iterations diminishes as we increase the value of p, with $p \geq 0.5$ leading to significantly less restarts on average.

Finally, we present the results for the Fletcher-Reeves formula (5), classically established as a variant with theoretical guarantees but less practical appeal [23]. As shown in Tables 6 and 7, this update leads to a much worse performance for all the NCG methods,

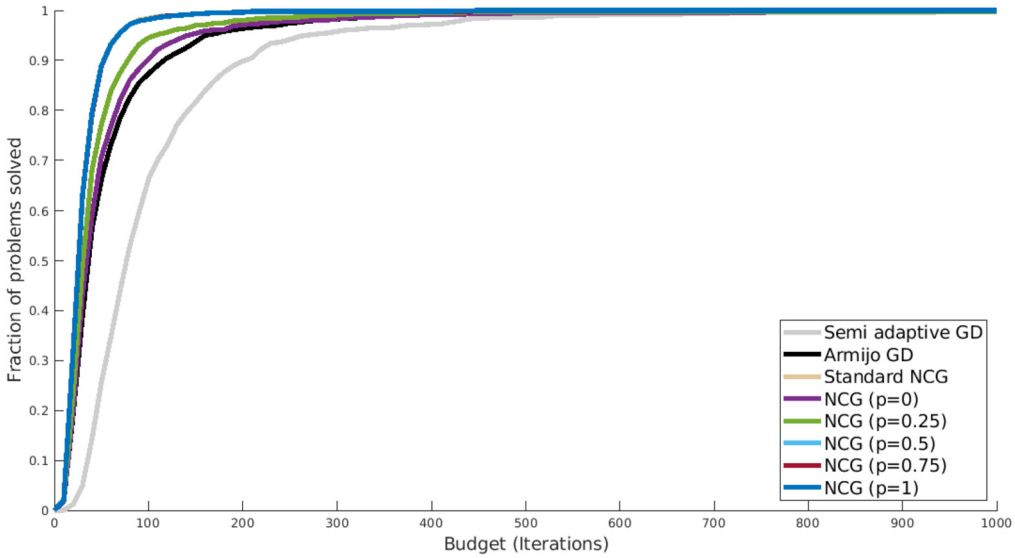

Fig. 4. Fraction of nonconvex problems with the Tukey biweight loss (27) solved as function of the iteration budget (truncated at 1000). The curves corresponding to Restarted NCG with $p \in \{0.5, 0.75, 1\}$ overlap with that of Standard NCG. All NCG variants use the PRP+ formula (7).

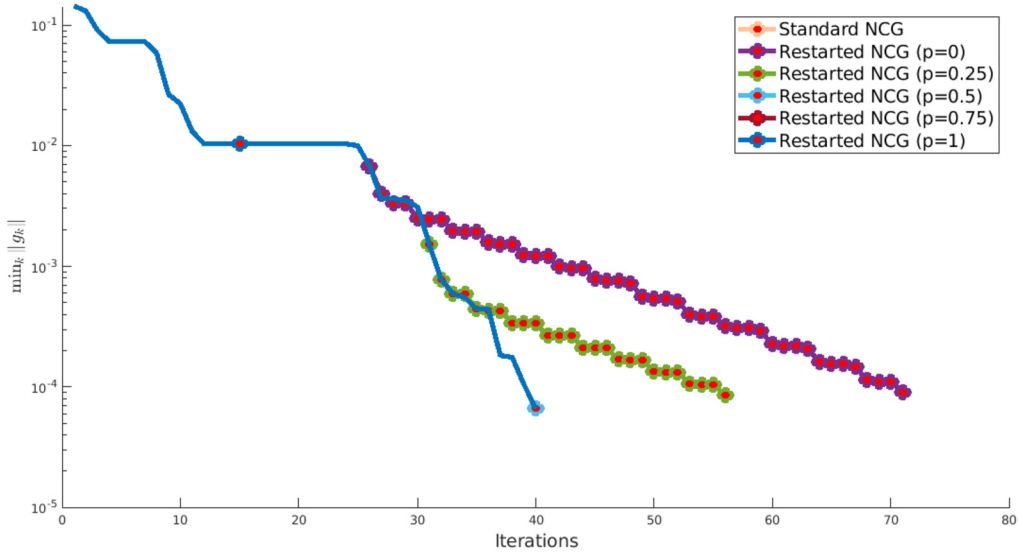

Fig. 5. Minimum gradient norm for a representative run using the smooth biweight loss (26) using the PRP+ formula (7). The curves corresponding to Restarted NCG with $p \in \{0.75, 1\}$ overlap with that of Standard NCG, and all have the same restarted iteration (red circles). Restarted NCG($p = 0.5$) had one extra restarted iteration, but overlaps with Standard NCG otherwise. Restarted NCG($p = 0.25$) variant had 27 restarted iterations, while Restarted NCG($p = 0$) had 47 restarted iterations.

Table 4
Statistics on 1000 random instances of robust linear regression using smoothed biweight loss (problem (26)) and a budget of 10000 iterations. All methods use the HZ formula (8).

Method	Problems solved	Avg. restart (%)
Standard NCG	1000	0.00%
NCG(0)	1000	52.8%
NCG(0.25)	1000	21.8%
NCG(0.5)	1000	0.56%
NCG(0.75)	1000	0.62%
NCG(1)	1000	0.76%

Table 5
Statistics on 1000 random instances of robust linear regression using Tukey loss (problem (27)) and a budget of 10000 iterations. All methods use the HZ formula (8).

Method	Problems solved	Avg. restart (%)
Standard NCG	1000	0.00%
NCG(0)	1000	48.5%
NCG(0.25)	1000	26.8%
NCG(0.5)	1000	1.28%
NCG(0.75)	1000	0.75%
NCG(1)	1000	0.86%

Table 6
Statistics on 1000 random instances of robust linear regression using smoothed biweight loss (problem (26)) and a budget of 10000 iterations. All variants of Algorithm 1 use $\beta_{k+1} = \beta_{k+1}^{FR}$.

Method	Problems solved	Avg. restart (%)
Standard NCG	9	0.03%
NCG(0)	122	2.98%
NCG(0.25)	197	0.94%
NCG(0.5)	216	0.02%
NCG(0.75)	368	0.03%
NCG(1)	514	0.03%

Table 7
Statistics on 1000 random instances of robust linear regression using Tukey loss (problem (27)) and a budget of 10000 iterations. All variants of Algorithm 1 use $\beta_{k+1} = \beta_{k+1}^{FR}$.

Method	Problems solved	Avg. restart (%)
Standard NCG	629	0.07%
NCG(0)	730	11.0
NCG(0.25)	759	4.59%
NCG(0.5)	769	0.11%
NCG(0.75)	839	0.06%
NCG(1)	876	0.07

and in that setting all methods with modified restart give better results than Standard NCG. Interestingly, Figs. 6 and 7 show that gradient descent actually outperforms the NCG variants, but that adding the modified restarted condition consistently improves

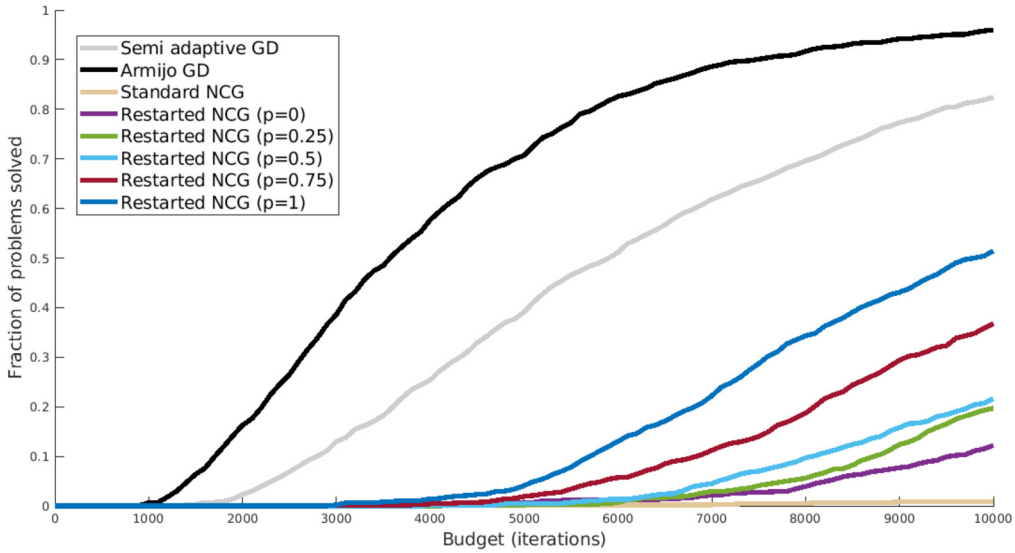

Fig. 6. Fraction of nonconvex problems with the smooth biweight loss (26) solved as function of the iteration budget. The curves corresponding to Restarted NCG with $p \in \{0.5, 0.75, 1\}$ overlap with that of Standard NCG. All NCG variants use the β_{k+1}^{FR} formula.

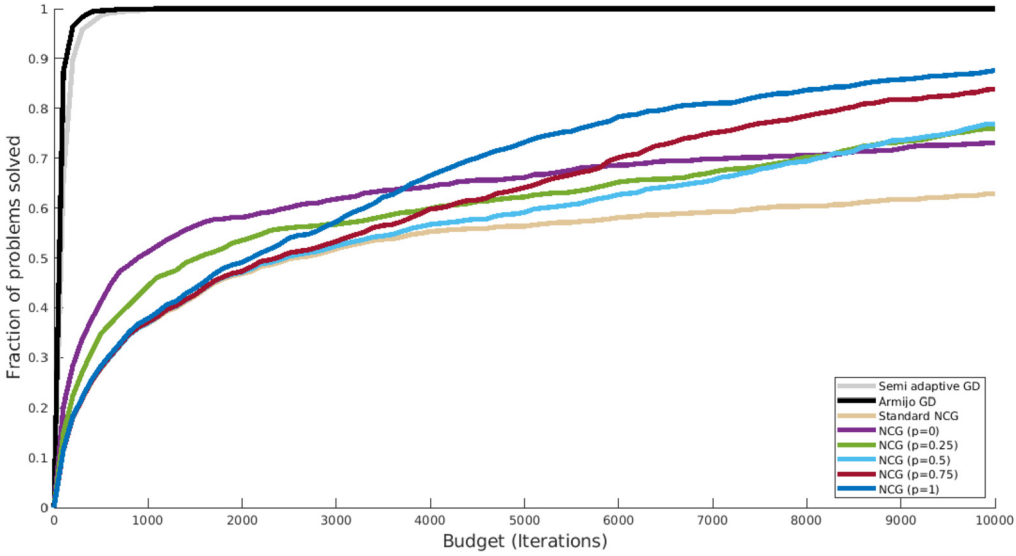

Fig. 7. Fraction of nonconvex problems with the Tukey biweight loss (27) solved as function of the iteration budget. The curves corresponding to Restarted NCG with $p \in \{0.5, 0.75, 1\}$ overlap with that of Standard NCG. All NCG variants use the β_{k+1}^{FR} formula.

the method's performance as p gets closer to 1. These results suggest that restarting conditions may be a way to improve the performance of a Fletcher-Reeves nonlinear conjugate gradient method.

Overall, our experiments indicate that our proposed framework can perform quite closely to a standard implementation of nonlinear CG on nonconvex regression problems. Setting $p \geq 0.5$ emerges as the best choice to track the behavior of standard nonlinear CG for certain parameter formulas, and even improve over it in the case of the FR formula. When the PRP+ formula is used, the restart condition is rarely triggered, thus the number of non-restarted iterations dominates that of restarted iterations. Our complexity guarantees in this (admittedly very specific) setting can therefore improve over the theoretically fastest methods based on accelerated gradient [4], and provide an ad-hoc justification for this practical behavior.

4.3. Smooth optimization benchmark

We now consider a more substantial test set consisting of CUTEst collection [18].[4] Our test comprises 149 smooth unconstrained optimization problems previously used to compare methods with and without complexity guarantees [12]. We chose the default dimension of each problem when indicated, or used the largest dimension below 1000 otherwise. The complete problem list is given in Table 8.

We only consider nonlinear conjugate gradient techniques in our comparison. We ran Standard NCG, Orthog NCG and Restarted NCG for $p \in \{0, 0.5, 0.75, 1\}$ using different values for the β_{k+1} parameter. All methods were run with a budget of 10000 iterations. A run was considered convergent whenever it reached a point x_k such that

$$\|\nabla f(x_k)\| \leq \epsilon \max\{1, \|\nabla f(x_0)\|\},$$

where $\epsilon = 10^{-5}$.

Results We present our results under the form of performance profiles [16] for three choices of formula for β_{k+1}. We provide those profiles using both the number of iterations (which would also correspond to the number of gradient evaluations) and the number of function evaluations as budget indicators.

When the FR formula (5) is used, we again observe that using $p \geq 0.5$ in Restarted NCG yields a profile quite close to that of Standard NCG. As illustrated by Figs. 8 and 9, the iteration and evaluation profiles are highly similar (yet not identical), suggesting that the number of backtracking line-search iterations is essentially constant.

Figs. 10 and 11 show profiles obtained with the HZ formula (8). On these plots, Standard NCG is capable of solving a larger fraction of the problems than the Restarted NCG methods. However, we still observe that the best profile among the Restarted NCG methods are obtained for $p \geq 0.5$.

[4] Downloaded from GitHub on June 8, 2021.

Table 8
List of CUTEst problems.

ALLINITU	4	ARGLINA	200	ARGLINB	200
ARGLINC	200	ARWHEAD	1000	BARD	3
BDEXP	100	BDQRTIC	100	BEALE	2
BIGGS6	6	BOX3	3	BRATU1D	77
BRKMCC	2	BROWNAL	10	BROWNBS	2
BROWNDEN	4	BROYDN7D	10	BRYBND	10
CHAINWOO	1000	CHNROSNB	10	CLIFF	2
CLPLATEA	49	CLPLATEB	49	CLPLATEC	49
COSINE	1000	CRAGGLVY	4	CUBE	2
CURLY10	1000	CURLY20	1000	CURLY30	1000
DECONVU	63	DENSCHNA	2	DENSCHNB	2
DENSCHNC	2	DENSCHND	3	DENSCHNE	3
DENSCHNF	2	DIXMAANA	15	DIXMAANB	15
DIXMAANC	15	DIXMAAND	15	DIXMAANE	15
DIXMAANF	15	DIXMAANG	15	DIXMAANH	15
DIXMAANI	15	DIXMAANJ	15	DIXMAANK	15
DIXMAANL	15	DIXON3DQ	10	DJTL	2
DQDRTIC	10	DQRTIC	500	EDENSCH	36
EG2	1000	EIGENALS	110	EIGENBLS	110
ENGVAL1	2	ENGVAL2	3	ERRINROS	50
EXPFIT	2	EXTROSNB	5	FLETCBV2	10
FLETCBV3	10	FLETCHBV	10	FLETCHCR	10
FMINSRF2	64	FMINSURF	16	FREUROTH	2
GENHUMPS	1000	GENROSE	500	GROWTHLS	3
GULF	3	HAIRY	2	HATFLDD	3
HATFLDE	3	HEART6LS	6	HEART8LS	8
HELIX	3	HILBERTA	10	HILBERTB	5
HIMMELBB	2	HIMMELBF	4	HIMMELBG	2
HIMMELBH	2	HUMPS	2	INDEF	1000
JENSMP	2	KOWOSB	4	LIARWHD	36
LOGHAIRY	2	MANCINO	100	MARATOSB	2
MEXHAT	2	MEYER3	3	MSQRTALS	4
MSQRTBLS	9	NONCVXU2	1000	NONCVXUN	1000
NONDIA	1000	NONDQUAR	100	NONMSQRT	9
OSBORNEA	5	OSBORNEB	11	PALMER1C	8
PALMER1D	7	PALMER1E	8	PALMER2C	8
PALMER2E	8	PALMER3C	8	PALMER3E	8
PALMER4C	8	PALMER4E	8	PALMER5C	6
PALMER5D	4	PALMER6C	8	PALMER7C	8
PALMER8C	8	PENALTY1	1000	PENALTY2	4
PENALTY3	50	PFIT1LS	3	PFIT2LS	3
PFIT3LS	3	PFIT4LS	3	POWER	10
QUARTC	25	ROSENBR	2	SCOSINE	1000
SCURLY10	1000	SCURLY20	1000	SCURLY30	1000
SINEVAL	2	SINQUAD	5	SISSER	2
SNAIL	2	SPARSINE	1000	SPMSRTLS	28
SROSENBR	10	STRATEC	10	TOINTQOR	50
TRIDIA	30	VARDIM	10	VAREIGVL	50
VIBRBEAM	8	WATSON	31	WOODS	4
YFITU	3	ZANGWIL2	2		

Finally, we provide results obtained using the PRP+ formula (7) through Figs. 12 and 13. The results are of the same flavor than the previous ones, although the gap between Standard NCG and the best Restarted NCG methods is smaller.

In a nutshell, these experiments suggest that the Restarted NCG variants may exhibit worse performance than a standard nonlinear conjugate gradient approach (that just restarts when it computes a non-descent search direction). The gap between a standard

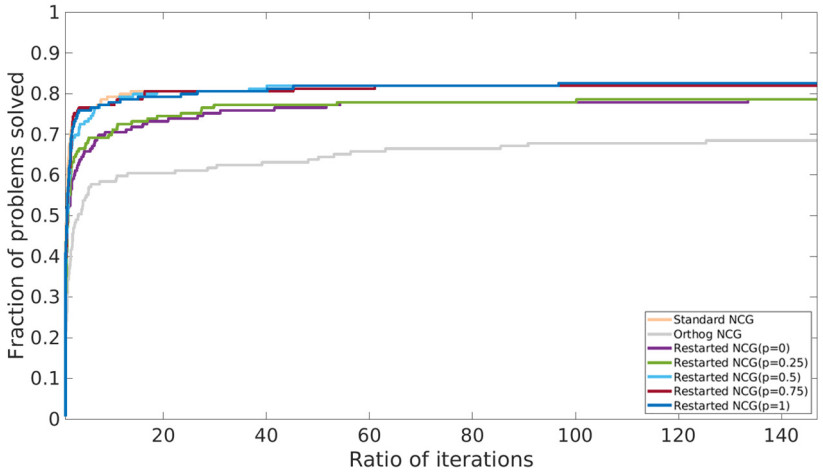

Fig. 8. Performance profiles (iterations) for nonlinear CG methods with the FR formula (5) formula on a benchmark from CUTEst.

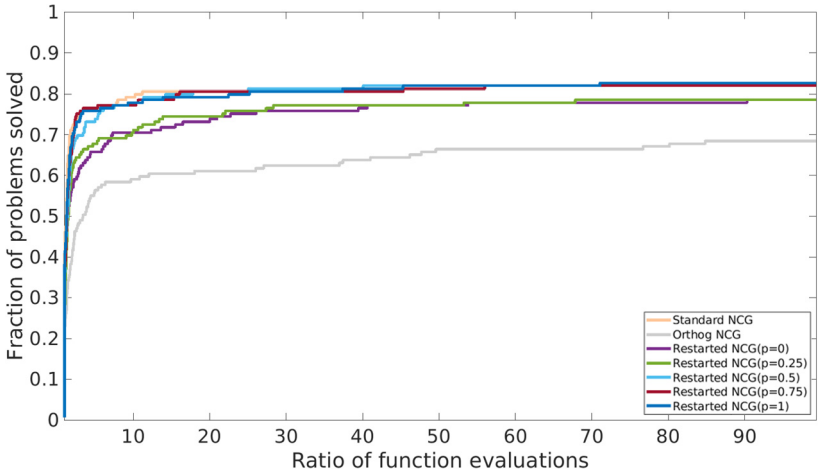

Fig. 9. Performance profiles (function evaluations) for nonlinear CG methods with the FR formula (5) formula on a benchmark from CUTEst.

method and our restarted variant can be partially explained by the additional requirements put in place to ensure complexity guarantees. However, our experiments with the Fletcher-Reeves formula also show that the restarting condition can have quite a minor effect on the performance, which is encouraging for future investigation on these aspects. Finally, we note that the Orthog NCG variant did not perform well on this test set, regardless of the formula that was used. Our interpretation is that the loss of orthogonality here, as measured by the test $|g_k^{\mathrm{T}} g_{k+1}| \geq \sigma \|g_k\|^2$ is not necessarily detrimental to the algorithm performance. Though out of the scope of this work, we conjecture that varying the power within the right-hand side of the condition might improve the performance.

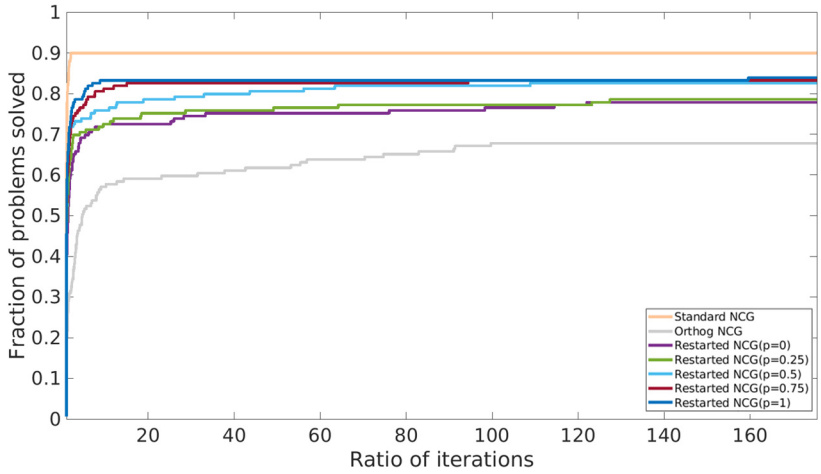

Fig. 10. Performance profiles (iterations) for nonlinear CG methods with the HZ formula (8) formula on a benchmark from CUTEst.

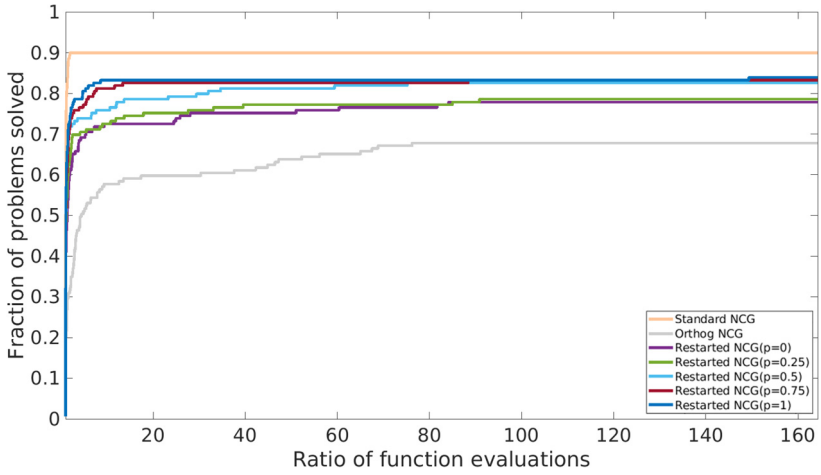

Fig. 11. Performance profiles (function evaluations) for nonlinear CG methods with the HZ formula (8) formula on a benchmark from CUTEst.

5. Conclusion

In this paper, we presented a nonlinear conjugate gradient framework based on Armijo line search and a modified restart condition, which we endowed with worst-case complexity guarantees. Although the results are of the same order than that for gradient descent, our complexity bound illustrates that better properties of nonlinear conjugate gradient may improve the overall number of iterations. Our motivation for considering this particular framework was the remarkable performance of Armijo PRP+ nonlinear CG on a nonconvex regression problem. Our experiments on such instances suggest that

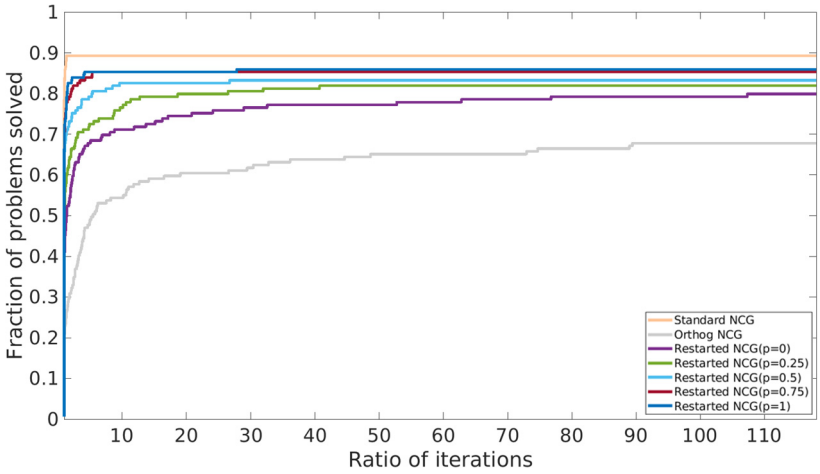

Fig. 12. Performance profiles (iterations) for nonlinear CG methods with the PRP+ formula (7) formula on a benchmark from CUTEst.

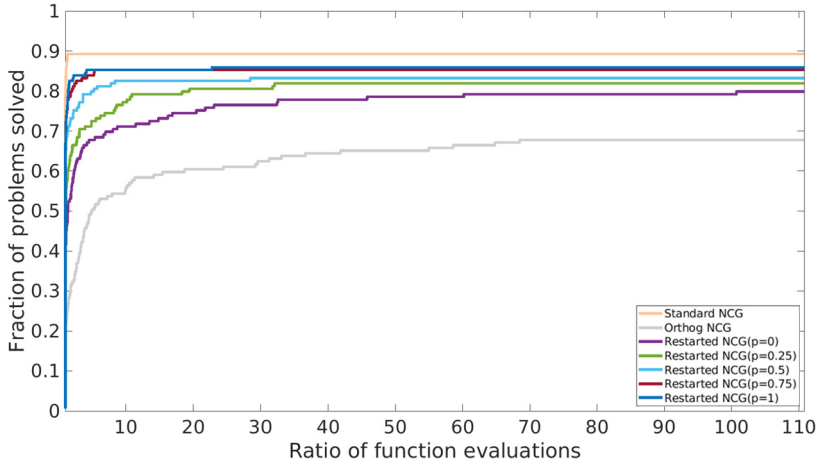

Fig. 13. Performance profiles (function evaluations) for nonlinear CG methods with the PRP+ formula (7) formula on a benchmark from CUTEst.

our new restart condition may be parameterized so as to match the original nonlinear CG method, thereby providing a theoretical justification of the performance of this method.

Our analysis does not leverage the specific definition of the search directions, but rather checks their properties a posteriori. A natural continuation of the present paper would consist in enforcing similar properties by design, possibly by using stronger line-search conditions such as strong Wolfe. In addition, the design of a nonlinear conjugate gradient with strictly better complexity bounds than gradient descent remains an open question. Recent advances in this area for convex optimization [27] might provide insights regarding the variants that are most amenable to a complexity analysis with guaranteed improvement over gradient descent.

Declaration of competing interest

No conflict of interest is to be reported.

References

[1] A.E. Beaton, J.W. Tukey, The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data, Technometrics 16 (1974) 147–185.
[2] E.G. Birgin, J.L. Gardenghi, J.M. Martínez, S.A. Santos, Ph.L. Toint, Worst-case evaluation complexity for unconstrained nonlinear optimization using high-order regularized models, Math. Program. 163 (2017) 359–368.
[3] E.G. Birgin, J.M. Martínez, The use of quadratic regularization with a cubic descent condition for unconstrained optimization, SIAM J. Optim. 27 (2017) 1049–1074.
[4] Y. Carmon, J.C. Duchi, O. Hinder, A. Sidford, "Convex until proven guilty": dimension-free acceleration of gradient descent on non-convex functions, in: Proceedings of the International Conference on Machine Learning, Sydney, Australia, August 2017, 2017, pp. 654–663.
[5] Y. Carmon, J.C. Duchi, O. Hinder, A. Sidford, Accelerated methods for non-convex optimization, SIAM J. Optim. 28 (2018) 1751–1772.
[6] Y. Carmon, J.C. Duchi, O. Hinder, A. Sidford, Lower bounds for finding stationary points II: first-order methods, Math. Program. 185 (2021) 315–355.
[7] C. Cartis, N.I.M. Gould, Ph.L. Toint, On the complexity of steepest descent, Newton's and regularized Newton's methods for nonconvex unconstrained optimization, SIAM J. Optim. 20 (2010) 2833–2852.
[8] C. Cartis, N.I.M. Gould, Ph.L. Toint, Adaptive cubic regularisation methods for unconstrained optimization. Part II: worst-case function- and derivative-evaluation complexity, Math. Program. 130 (2011) 295–319.
[9] C. Cartis, N.I.M. Gould, Ph.L. Toint, Optimal Newton-type methods for nonconvex optimization, Technical Report naXys-17-2011, Department of Mathematics, University of Namur, Belgium, 2011.
[10] C. Cartis, N.I.M. Gould, Ph.L. Toint, Worst-case evaluation complexity and optimality of second-order methods for nonconvex smooth optimization, in: Proceedings of the International Congress of Mathematicians (ICM 2018), vol. 3, 2019, pp. 3697–3738.
[11] C. Cartis, Ph.R. Sampaio, Ph.L. Toint, Worst-case evaluation complexity of non-monotone gradient-related algorithms for unconstrained optimization, Optimization 64 (2015) 1349–1361.
[12] F.E. Curtis, D.P. Robinson, C.W. Royer, S.J. Wright, Trust-region Newton-CG with strong second-order complexity guarantees for nonconvex optimization, SIAM J. Optim. 31 (2021) 518–544.
[13] F.E. Curtis, D.P. Robinson, M. Samadi, A trust region algorithm with a worst-case iteration complexity of $O\left(\epsilon^{-3/2}\right)$ for nonconvex optimization, Math. Program. 162 (2017) 1–32.
[14] F.E. Curtis, D.P. Robinson, M. Samadi, An inexact regularized Newton framework with a worst-case iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$ for nonconvex optimization, IMA J. Numer. Anal. 39 (2019) 1296–1327.
[15] Y.-H. Dai, Conjugate gradient methods with Armijo-type line searches, Acta Math. Appl. 18 (2002) 123–130.
[16] E.D. Dolan, J.J. Moré, Benchmarking optimization software with performance profiles, Math. Program. 91 (2002) 201–213.
[17] J.C. Gilbert, J. Nocedal, Global convergence properties of conjugate gradient methods for optimization, SIAM J. Optim. 2 (1992) 21–42.
[18] N.I.M. Gould, D. Orban, Ph.L. Toint, CUTEst: a constrained and unconstrained testing environment with safe threads, Comput. Optim. Appl. 60 (2015) 545–557.
[19] L. Grippo, S. Lucidi, A globally convergent version of the Polak-Ribière conjugate gradient method, Math. Program. 78 (1997) 375–391.
[20] L. Grippo, S. Lucidi, Convergence conditions, line search algorithms and trust region implementations for the Polak-Ribière conjugate gradient method, Optim. Methods Softw. 20 (2005) 71–98.
[21] W.W. Hager, H. Zhang, A new conjugate gradient method with guaranteed descent and an efficient line search, SIAM J. Optim. 16 (2005) 170–192.
[22] W.W. Hager, H. Zhang, Algorithm 851: CG_DESCENT, a conjugate gradient method with guaranteed descent, ACM Trans. Math. Softw. 32 (2006) 113–137.

[23] W.W. Hager, H. Zhang, A survey of nonlinear conjugate gradient methods, Pac. J. Optim. 2 (2006) 35–58.
[24] S. Karimi, S.A. Vavasis, A unified convergence bound for conjugate gradient and accelerated gradient, arXiv:1605.00320, 2016.
[25] S. Karimi, S.A. Vavasis, A single potential governing convergence of conjugate gradient, accelerated gradient and geometric descent, arXiv:1712.09498, 2017.
[26] S. Karimi, S.A. Vavasis, Detecting and correcting the loss of independence in nonlinear conjugate gradient, arXiv:1202.1479v2, 2018.
[27] S. Karimi, S.A. Vavasis, Nonlinear conjugate gradient for smooth convex functions, arXiv:2111.11613, 2021.
[28] J.J. Moré, S.M. Wild, Benchmarking derivative-free optimization algorithms, SIAM J. Optim. 20 (2009) 172–191.
[29] A.S. Nemirovski, D.B. Yudin, Problem Complexity and Method Efficiency in Optimization, John Wiley & Sons, New York, 1983.
[30] Yu. Nesterov, B.T. Polyak, Cubic regularization of Newton method and its global performance, Math. Program. 108 (2006) 177–205.
[31] J. Nocedal, S.J. Wright, Numerical Optimization, second edition, Springer Ser. Oper. Res. Financ. Eng., Springer-Verlag, New York, 2006.
[32] R. Pytlak, Conjugate Gradient Algorithms in Nonconvex Optimization, Nonconvex Optimization and Its Applications, vol. 89, Springer-Verlag, Berlin Heidelberg, 2009.
[33] C.W. Royer, M. O'Neill, S.J. Wright, A Newton-CG algorithm with complexity guarantees for smooth unconstrained optimization, Math. Program. 180 (2020) 451–488.
[34] C.W. Royer, S.J. Wright, Complexity analysis of second-order line-search algorithms for smooth nonconvex optimization, SIAM J. Optim. 28 (2018) 1448–1477.
[35] Ph.L. Toint, Nonlinear stepsize control, trust regions and regularizations for unconstrained optimization, Optim. Methods Softw. 28 (2013) 82–95.
[36] S.J. Wright, Optimization algorithms for data analysis, in: A.C. Gilbert, M.W. Mahoney, J.C. Duchi (Eds.), The Mathematics of Data, in: IAS/Park City Mathematics Series, vol. 25, AMS, IAS/Park City Mathematics Institute, and Society for Industrial and Applied Mathematics, Princeton, 2018.