A line-search algorithm inspired by the adaptive cubic regularization framework, with a worst-case complexity $O(\epsilon^{-3/2})$.

E. Bergou * Y. Diouane† S. Gratton‡

June 16, 2017

Abstract

Adaptive regularized framework using cubics (ARC) has emerged as an alternative to line-search and trust-region for smooth nonconvex optimization, with an optimal complexity amongst second-order methods. In this paper, we propose and analyze the use of a special (iteration dependent) scaled norm in ARC of the form $\|x\|_M = \sqrt{x^T M x}$ for $x \in \mathbb{R}^n$, where $M$ is a symmetric positive definite matrix satisfying specific secant equation. Within the proposed norm, ARC behaves as a line-search algorithm along the Newton direction, with a special backtracking strategy and acceptability condition. The obtained line-search algorithm enjoys the same convergence and complexity properties as ARC, in particular the complexity for finding approximate first-order stationary points is $O(\epsilon^{-3/2})$. Furthermore, we have proposed similar analysis when considering the trust-region framework. The good potential of the new line-search algorithms is showed on a set of large scale optimization problems.

Keywords: Nonlinear optimization, unconstrained optimization, line-search methods, adaptive regularized framework using cubics, trust-region methods, worst-case complexity.

1 Introduction

We consider the following unconstrained optimization problem:

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

where $f : \mathbb{R}^n \to \mathbb{R}$ is a given smooth function known as the objective function. Classical iterative methods for solving (1) are trust-region (TR) [8, 18] and line-search (LS) [10] algorithms. Recently an adaptive cubic regularization framework with cubics (ARC) has been proposed by Cartis et al [5].

The worst-case evaluation complexity of finding an $\epsilon$-approximate first-order critical point using TR or LS methods is shown to be computed in at most $O(\epsilon^{-2})$ function or gradient

\*MaIAGE, INRA, Université Paris-Saclay, 78350 Jouy-en-Josas, France (elhoucine.bergou@inra.fr).
†Institut Supérieur de l’Aéronautique et de l’Espace (ISAE-SUPAERO), Université de Toulouse, 31055 Toulouse Cedex 4, France (youssef.diouane@isae.fr).
‡INP-ENSEEIHT, Université de Toulouse, 31071 Toulouse Cedex 7, France (serge.gratton@enseeiht.fr).
evaluations, where $\epsilon \in (0, 1)$ is a user-defined accuracy threshold on the gradient norm $[16, 14, 7]$. ARC takes at most $O(\epsilon^{-3/2})$ to reduce the gradient norm below $\epsilon$, and thus it is improving substantially the worst-case complexity over the classical TR/LS methods $[4]$. Such complexity bound can be improved using higher order regularized models, we refer the reader for instance to the references $[2, 6]$. Recently, a non-standard TR method $[9]$ is proposed with the same worst-case complexity bound as ARC. It is proved also that the same worst-case complexity $O(\epsilon^{-3/2})$ can be achieved by mean of a specific variable-norm in a TR method $[15]$ or using quadratic-regularization $[3]$. All previous approaches use a cubic sufficient-descent condition instead of the more usual predicted-reduction based descent. Generally, they need to solve more than one linear system in sequence at each outer iteration, this makes the computational cost per iteration expensive. In this work we will derive an algorithm with the same worst-case complexity of $O(\epsilon^{-3/2})$ without imposing any cubic sufficient-descent condition on the objective function and requiring only to solve (approximately) one linear system per iteration to get a trial point.

In $[1]$ it has been shown how to use the so-called energy norm in the ARC/TR framework when a symmetric positive definite (SPD) approximation of the objective function Hessian is available. Within the energy norm, ARC/TR methods behave as LS algorithms along the Newton direction, with a special backtracking strategy and an acceptability condition in the spirit of ARC/TR methods. As far as the model of the objective function is convex (i.e., the approximate Hessian is SPD), the authors in $[1]$ developed a methodology to make ARC behaves as an LS algorithm and showed that the resulting algorithm enjoys the same convergence and complexity analysis properties as ARC, in particular the first-order complexity bound of $O(\epsilon^{-3/2})$. In the complexity analysis of ARC method $[4]$, it is required that the Hessian approximation has to approximate accurately enough the true Hessian (see $[4$, Assumption AM.4$]$), obtaining such convex approximation may be out of reach when handling nonconvex optimization. This paper generalizes the proposed methodology in $[1]$ to handle nonconvex models.

In this paper we show that by using a special scaled norm the computational cost of the ARC subproblem is nearly the same as solving a linear system. In fact, the proposed approach consists of the use of a scaled norm to define the regularization term in ARC subproblem. The chosen scaled norm is of the form $\|x\|_M = \sqrt{x^T M x}$ for $x \in \mathbb{R}^n$, where $M$ is an SPD matrix that satisfies a specific secant equation. At each iteration we choose a possible different matrix $M$. (So, $M = M_k$ where the subscript $k$ is the iteration index.) The specific choice of $M$ trivializes the solution of ARC subproblem (as it renders the ARC subproblem solution collinear with the Newton step) and hence leads to a class of LS algorithms, along the Newton direction, with a specific acceptability condition and that enjoy the same convergence and worst-case complexity results as the classical ARC method. Compared to ARC (when using the $l_2$-norm to define the regularization term in the subproblem), the dominant computational cost of the resulting algorithm is mainly the cost of successful iterations as the unsuccessful ones are getting inexpensive. To our knowledge, it is the first LS algorithm that enjoys the same convergence and complexity results (in a straightforward way) as in the classical ARC $[5, 4]$.

Additionally, we also consider a large-scale variant for the case where matrix factorizations are not affordable, implying that only iterative methods for computing a trial step can be used. Similar analysis is applied to the TR framework, where using the same scaled norm (as in ARC) we show that TR method can behave as an LS algorithm. Numerical illustrations (over a test set of large scale optimization problems) are given in order to assess the efficiency of the obtained LS algorithms.
We organize this paper as follows. In Section 2, we introduce the ARC method using a general scaled norm. Section 3 analyses the minimization of the cubic model and discusses the choice of the scaled norm that simplifies solving the ARC subproblem. Section 4 derives the new LS algorithm on the base of ARC (when the proposed scaled norm is used) and discusses how the iteration dependent $M$-norm can chosen so as to be uniformly equivalent to the Euclidean norm. We end the section by stating the necessary stopping criteria that is needed to maintain the complexity of ARC when the subproblem is solved iteratively. Similarly to ARC and using the same scaled norm, an LS algorithm in the spirit of TR algorithm is proposed in Section 5. Numerical tests are illustrated and discussed in Section 6. Conclusions and future improvements are given in Section 7.

Throughout this paper $\| \cdot \|$ will denote the vector or matrix $l_2$-norm. $\| \cdot \|_M$ will denote the scaled norm which is of the form $\| x \|_M = \sqrt{x^T M x}$ for $x \in \mathbb{R}^n$, where $M$ is a given SPD matrix. We denote also by $\text{sgn}(\alpha)$ the sign of a real $\alpha$.

## 2 The ARC framework

At a given iterate $x_k$, we define $m_k^Q : \mathbb{R}^n \rightarrow \mathbb{R}$ as an approximate second-order Taylor approximation of the objective function $f$ around $x_k$, i.e.,

$$m_k^Q(s) = f(x_k) + s^T g_k + \frac{1}{2} s^T B_k s,$$

where $g_k = \nabla f(x_k)$ is the gradient of $f$ at the current iterate $x_k$, and $B_k$ is a symmetric local approximation of the Hessian of $f$ at $x_k$. For ARC [5], the trial step $s_k$ is computed as an approximation to the global minimizer of the cubic model $m_k^C(s)$:

$$s_k^{ARC} = \arg \min_{s \in \mathbb{R}^n} m_k^C(s) = m_k^Q(s) + \frac{1}{3} \sigma_k \| s \|_M^3,$$

where $M_k$ is a positive definite matrix, and $\sigma_k > 0$ is a dynamic positive parameter that might be regarded as the reciprocal of the TR radius in TR algorithms (see [5]). The parameter $\sigma_k$ is taking into account the agreement between the objective function $f$ and the model $m_k^C$. To decide whether the trial step is acceptable or not a ratio between the actual reduction and the predicted reduction is computed, as follows:

$$\rho_k = \frac{f(x_k) - f(x_k + s_k^{ARC})}{f(x_k) - m_k^Q(s_k^{ARC})}.$$  (4)

For a given scalar $0 < \eta < 1$, the $k^{th}$ outer iteration will be said successful if $\rho_k \geq \eta$, and unsuccessful otherwise. For all successful iterations we set $x_{k+1} = x_k + s_k^{ARC}$; otherwise the current iterate is kept unchanged $x_{k+1} = x_k$. We note that, unlike ARC in [5, 4] where the cubic model is used to evaluate the denominator in (4), only the quadratic approximation ($m_k^Q$) is used in the comparison with the actual value of $f$ without the regularization parameter playing any role (as in [2]). Algorithm 1 gives a detailed description of ARC.

To ensure the global convergence of Algorithm 1, the trial step is required to provide a decrease greater than or equal to the reduction attained by the so-called Cauchy step which is defined as follows: $s_k^{ARC, \text{Cauchy}} = -\alpha_k^{\text{Cauchy}} g_k$, where $\alpha_k^{\text{Cauchy}} = \arg \min_{t > 0} m_k^C(-tg_k)$. 

3
Algorithm 1: ARC algorithm.

**Data:** select an initial point $x_0$ and the constant $0 < \eta < 1$. Set the initial regularization $\sigma_0 > 0$, and the constants $0 < \nu_1 \leq 1 < \nu_2$.

**for** $k = 1, 2, \ldots$ **do**

**Step 1:** compute the model gradient $g_k$ and an approximated Hessian $B_k$;

**Step 2:** compute the step $s_k^{\text{ARC}}$ as an approximate solution of (3);

**Step 3:** if $\rho_k \geq \eta$ then

- set $x_{k+1} = x_k + s_k^{\text{ARC}}$ and $\sigma_{k+1} = \nu_1 \sigma_k$;

else

- set $x_{k+1} = x_k$ and $\sigma_{k+1} = \nu_2 \sigma_k$;

**end**

**end**

In this case and under appropriate assumptions, convergence results of Algorithm 1 can be found in [5, 2]. Moreover, the algorithm ensures convergence to first-order stationary points with a steepest-descent-like function-evaluation complexity bound which is of order $\epsilon^{-2}$ to guarantee

$$\|g_k\| \leq \epsilon,$$

(5)

where $\epsilon > 0$ is pre-defined constant. By imposing a more stringent termination condition on the computation of the trial step $s_k^{\text{ARC}}$, one can improve function-evaluation complexity to be of the order of $\epsilon^{-3/2}$ to ensure (5) for the ARC algorithm (see [4, 2]). Such a termination condition is

$$\|\nabla m_k^C(s_k^{\text{ARC}})\| \leq \zeta \|s_k^{\text{ARC}}\|^2,$$

(6)

for some constant $\zeta > 0$ chosen at the start of the algorithm.

3 On the cubic model minimization

In this section, we will mostly focus on the solution of the subproblem (3) for a given outer iteration $k$. Thus the subscript $k$ will be dropped to keep the notations simple. In a precedent work [1], when the matrix $B$ is assumed to be positive definite, we showed that the minimizer $s_k^{\text{ARC}}$ of the cubic model defined in (3) is getting collinear with the Newton direction when the matrix $M$ is set to be equal to $B$. In this section we generalize our proposed approach to cover the case where the matrix $B$ still non-singular but not necessarily positive. We will explicit the condition to impose on the matrix $M$ in order to get the solution of the ARC subproblem at a modest computational cost.

3.1 Exact solution of the ARC subproblem

For the purpose of this subsection, the vector $s_k^Q$ will denote the exact solution of the linear system $Bs = -g$, i.e.,

$$s_k^Q = -B^{-1}g,$$

(7)

and $d_N$ denotes the Newton direction, i.e., $d_N = -\frac{s_k^Q}{s_k^Q s_k^Q}$. Note that the direction $d_N$ is a descent direction for the objective function $f$ (i.e. $d_N g < 0$), and that when the matrix $B$ is positive
definite, \( s^Q \) corresponds to the exact solution of the unconstrained quadratic optimization: 
\[
\min_{s \in \mathbb{R}^n} m^Q(s).
\]
In the sequel, we will assume that the gradient \( g \) is non zero (i.e., Algorithm 1 has not reached convergence yet), and that the approximate Hessian matrix \( B \) is non-singular. We will assume also that the Newton direction is not orthogonal with the gradient, in order to ensure sufficient descent condition on that direction.

**Assumption 3.1** For all outer iterations, the gradient \( g \) is non null (i.e., \( g \neq 0 \)), the approximate Hessian matrix \( B \) of the model is non-singular, and 
\[
|g^\top s^Q| \geq \epsilon_d \|g\| \|s^Q\| \quad \text{where} \quad \epsilon_d > 0
\]
is a pre-defined positive constant.

The assumptions “\( B \) non-singular” and “\( |g^\top s^Q| \geq \epsilon_d \|g\| \|s^Q\| \)” are needed in our analysis in order to ensure the existence of the Newton direction and that the latter is a sufficient descent direction for the objective function. When such assumption is violated one can perturb (in a minimal way) the matrix \( B \) using regularization techniques. Furthermore, close to the solution the Hessian is usually SPD and thus such modification may be only performed “far” from the optimal solution.

In what comes next, we define \( s^\text{ARC} \) as the Newton-like step associated with the minimization of the cubic model \( m^C \):
\[
s^\text{ARC} = \alpha_d N d_N, \quad \text{where} \quad \alpha_d = \arg \min_{t > 0} m^C_k(t d_N).
\]  
(8)

The next theorem gives the explicit form of \( s^\text{ARC} \).

**Theorem 3.1** Let Assumption 3.1 holds. The Newton-like step (8) is given by:
\[
s^\text{ARC} = \delta^\text{ARC} s^Q, \quad \text{where} \quad \delta^\text{ARC} = 2 \frac{1}{1 - \sgn(g^\top s^Q) \sqrt{2 + 4 \sigma \|s^Q\|_M^3 \|g^\top s^Q\|_M^3}} \quad \text{and} \quad \alpha_d = \frac{1}{g^\top s^Q} + \frac{\sqrt{2 + 4 \sigma \|s^Q\|_M^3 \|g^\top s^Q\|_M^3}}{2 \sigma \|s^Q\|_M^3 \|g^\top s^Q\|_M^3} \quad \text{and} \quad \epsilon_d > 0.
\]  
(9)

**Proof.** Indeed, for all \( t > 0 \), one has
\[
m^C(t d_N) - m^C(0) = t g^\top d_N + \frac{t^2}{2} d_N^\top B d_N + \frac{\sigma t^3}{3} d_N = -t - \frac{1}{g^\top s^Q} t^2 + \frac{\sigma \|s^Q\|_M^3 t^3}{3}.
\]  
We compute the value of the parameter \( t \) at which the unique minimizer of the above function is attained. Let \( \alpha_d \) denotes this optimal parameter. Taking the derivative of (10) with respect to \( t \) and equating the result to zero, one gets
\[
0 = -1 - \frac{1}{g^\top s^Q} \alpha_d + \frac{\sigma \|s^Q\|_M^3}{\|g^\top s^Q\|_M^3} (\alpha_d)^2,
\]  
(10)

and thus, since \( \alpha_d > 0 \),
\[
\alpha_d = \frac{1}{g^\top s^Q} + \frac{\sqrt{2 + 4 \sigma \|s^Q\|_M^3 \|g^\top s^Q\|_M^3}}{2 \sigma \|s^Q\|_M^3 \|g^\top s^Q\|_M^3} = \frac{2 g^\top s^Q}{1 - \sgn(g^\top s^Q) \sqrt{2 + 4 \sigma \|s^Q\|_M^3 \|g^\top s^Q\|_M^3}}.
\]
Hence, $s^\text{ARC}_N = \delta^\text{ARC}_N s^Q_N$, where $\delta^\text{ARC}_N = \frac{2}{1 - \text{sgn}(g^\top s^Q_N) \sqrt{1 + 4 \frac{s^Q_N \| s^Q_N \|_M^2}{g^\top s^Q_N}}}$.

In general, the matrix $M$ can be chosen arbitrarily as long as it is an SPD matrix. Our goal, in this paper, is to determine how we can choose the matrix $M$ so that the Newton-like step $s^\text{ARC}_N$ becomes a stationary point of subproblem (3). The following theorem gives explicitly the necessary and sufficient condition on the matrix $M$ to reach this aim.

**Theorem 3.2** Let Assumption 3.1 holds. The step $s^\text{ARC}_N$ is a stationary point for the subproblem (3) if and only if there exists $\theta > 0$ such that $Ms^Q_N = \frac{\theta}{g^\top s^Q_N} g$. Note that in this case $\theta = \|s^Q_N\|_M^2$.

**Proof.** Indeed, if we suppose that the step $s^\text{ARC}_N$ is a stationary point of the subproblem (3), this means that

$$\nabla s m^C(s^\text{ARC}_N) = g + B s^\text{ARC}_N + \|s^\text{ARC}_N\|_M M s^\text{ARC}_N = 0,$$

In another hand, $s^\text{ARC}_N = \alpha_N d_N$ where $d_N = \frac{B^{-1}g}{g^\top s^Q_N}$ and $\alpha_N > 0$ solution of $-1 - \frac{1}{\|s^Q_N\|_M^2} \alpha_N + \frac{\alpha_N}{\|s^Q_N\|_M^2} (\alpha_N)^2 = 0$. Substituting $s^\text{ARC}_N$ in (11), we obtain that

$$0 = \nabla s m^C(s^\text{ARC}_N) = g + \frac{\alpha_N}{g^\top s^Q_N} g + \|s^\text{ARC}_N\|_M MB^{-1} g = \left(1 + \frac{\alpha_N}{g^\top s^Q_N}\right) g + \frac{\alpha_N}{\|s^Q_N\|_M^2} (\alpha_N)^2 \left(\frac{g^\top s^Q_N}{\|s^Q_N\|_M^2} MB^{-1} g\right) = \left(\frac{\alpha_N}{\|s^Q_N\|_M^2} (\alpha_N)^2\right) \left(g - \frac{g^\top s^Q_N}{\|s^Q_N\|_M^2} MB^{-1} g\right).$$

Equivalently, we conclude that $Ms^Q_N = \frac{\theta}{g^\top s^Q_N} g$ where $\theta = \|s^Q_N\|_M^2 > 0$.

The key assumption to ensure that the ARC subproblem stationary point is equal to the Newton-like step $s^\text{ARC}_N$, is the choice of the matrix $M$ which satisfies the following secant-like equation $Ms^Q_N = \frac{\theta}{g^\top s^Q_N} g$ where $s^Q_N$ is given by (7) and $\theta > 0$. The existence of such matrix $M$ is not problematic as far as Assumption 3.1 holds. For instance, for a fixed value of $\theta > 0$, one may use BFGS techniques to find a matrix $M$. Note that in the formula of $s^\text{ARC}_N$ the matrix $M$ is used only through the computation of the $M$-norm of $s^Q_N$. Therefore the explicit formula of the matrix $M$ is not needed, and only the value of $\theta = \|s^Q_N\|_M^2$ suffices for the computations. In what comes next, we will impose the following assumption:

**Assumption 3.2** The iteration dependent matrix $M$ is SPD and it satisfies $Ms^Q_N = \frac{\theta}{g^\top s^Q_N} g$ where $\theta > 0$ is given and $s^Q_N = -B^{-1} g$.

The following theorem gives a sufficient condition to ensure that $s^\text{ARC}_N$ is the unique minimizer of $m^C$.

**Theorem 3.3** Let Assumptions 3.1 and 3.2 hold. If the matrix $B + \|s^\text{ARC}_N\|_M M$ is positive definite, then the step $s^\text{ARC}_N$ is the unique minimizer for the subproblem (3).
Proof. Indeed, using [8, Theorem 3.1], we have that, for a given vector \( s_* \), it is a global minimizer of \( m^C \) if and only if it satisfies
\[
(B + \lambda_* M)s_* = -g
\]
where \( B + \lambda_* M \) is positive semidefinite matrix and \( \lambda_* = \sigma \| s_* \|_M \). Moreover, if \( B + \lambda_* M \) is positive definite, \( s_* \) is unique.

Since \( Ms_* = \frac{\theta}{g^T s_*} g \), by applying Theorem 3.2, we see that
\[
(B + \lambda_* M)s_* = -g
\]
with \( \lambda_* = \sigma \| s_*^{ARC} \|_M \). Thus, if we assume that \( B + \lambda_* M \) is positive definite matrix, then \( s_*^{ARC} \) is the unique global minimizer of the subproblem (3).

Under Assumptions 3.1 and 3.2, Theorem 3.3 states that the step \( s_*^{ARC} \) can be the global minimizer of the cubic model \( m^C \) as far as the matrix \( B + \lambda_* M \) is positive definite, where \( \lambda_* = \sigma \| s_*^{ARC} \|_M \). Note that
\[
\lambda_* = \sigma \| s_*^{ARC} \|_M = \frac{2\sigma \| s_*^{Q} \|_M}{1 - \sigma \| s_*^{Q} \|_M} \sqrt{1 + \frac{4 \sigma \| s_*^{Q} \|_M^2}{|g^T s_*^{Q}|}} \rightarrow +\infty \quad \text{as} \quad \sigma \rightarrow \infty.
\]
Thus, since \( M \) is an SPD matrix and the regularization parameter \( \sigma \) is increased for unsuccessful iterations in Algorithm 1, the positive definiteness of matrix \( B + \lambda_* M \) is guaranteed after finitely many unsuccessful iterations.

Another important practical consequence of Theorem 3.3 is that solving exactly the ARC subproblem amounts to computing the solution of the linear system \( Bs = -g \) and evaluating the scalar in (9). This means that solving the subproblem, while varying the value of \( \sigma \), is inexpensive once the solution of \( Bs = -g \) is found. This remark will be essential when considering unsuccessful steps in the overall optimization algorithm. Another consequence is that the subproblem solution can be obtained by using a direct method based on the Cholesky factorization provided that \( B \) is not too large for such an approach. In the next section we consider the case where \( Bs = -g \) is solved iteratively.

### 3.2 Approximate solution using subspace methods

For large scale optimization problems, computing \( s_*^{Q} \) can be prohibitively computationally expensive. We will show that it will be possible to relax this requirement by letting the step \( s_*^{Q} \) be only an approximation of the exact solution.

For an approximate solution, as far as the global convergence of Algorithm 1 is concerned, all what we need is that the solution of the subproblem (3) yields a decrease in the cubic model which is as good as a fraction of the Cauchy decrease. The latter is obtained by minimizing the model along the negative gradient direction [5]. In practice, a version of Algorithm 1 solely based on the Cauchy step would suffer from the same drawbacks as the steepest descent algorithm on ill-conditioned problems and faster convergence can be expected if the matrix \( B \) influences also the minimization direction. The main idea consists of achieving a further decrease on the cubic model (better that the Cauchy decrease) by projection onto a sequence of embedded Krylov
subspaces. We now show how to use a similar idea to compute a solution of the subproblem that is cheap to compute and yields a global convergence of Algorithm 1.

A classical way to approximate the exact solution $s^Q$ is by using subspace methods (typically a Krylov subspace method). For that, let $L_i$ be a subspace of $\mathbb{R}^n$ and $l$ its dimension. Let $Q_i$ denotes an $n \times l$ matrix whose columns form a basis of $L_i$. Thus for all $s \in L_i$, we have $s = Q_i z$, for some $z \in \mathbb{R}^l$. We will denote by $s^Q_i$ the exact stationary point of the model function $m^Q$ over the subspace $L_i$:

$$s^Q_i = -Q_i (Q_i^\top B Q_i)^{-1} Q_i^\top g.$$  

(12)

We define $s^{ARC}_{N,i}$ as the projection, onto the subspace $L_i$, of the step $s^{ARC}_N$ associated with the subproblem (3), that is $s^{ARC}_N = Q_i z_N$ where $z_N$ is the Newton-like step associated to the following reduced subproblem:

$$\min_{z \in \mathbb{R}^l} f(x) + z^\top Q_i^\top g + \frac{1}{2} z^\top Q_i^\top B Q_i z + \frac{1}{3} \sigma \|z\|_Q^3 M Q_i.$$  

(13)

We update Assumption 3.1 when only an approximate solution (in the subspace $L_i$) of the linear system $B s = g$ is available:

**Assumption 3.3** For all outer iterations, the gradient function $g$ is non null (i.e., $g \neq 0$), the approximate Hessian matrix $Q_i^\top B Q_i$ of the reduced model is non-singular, and that $|g^\top s^Q_i| > \epsilon_d \|g\|_M s^Q_i$, where $\epsilon_d > 0$ is a pre-defined positive constant.

In Theorem 3.3, using a specific $M$-norm, we showed that the global minimizer of the cubic model $m^C$ is collinear to the Newton-like step. We will now show that, as far as the matrix $M$ satisfies an appropriate assumption, $s^{ARC}_N$ will enjoy a similar collinearity property as in Theorem 3.3.

**Theorem 3.4** Let Assumption 3.3 holds. Then the

$$s^{ARC}_{N,i} = \delta^{ARC}_{N,i} s^Q_i,$$

where

$$\delta^{ARC}_{N,i} = \frac{2}{1 - \sgn(g^\top s^Q_i) \sqrt{1 + 4 \frac{\sigma \|z^Q_i\|^3}{|g| s^Q_i} |Q_i^\top g|}.$$  

(14)

**Proof.** Indeed, one has $s^{ARC}_{N,i} = Q_i z_N$, where $z_N$ is the Newton-like step associated to the reduced subproblem (13). Hence by applying Theorem 3.1 to the reduced subproblem (13), it follows that

$$z_N = \delta^{ARC}_N z^Q_*$$

where

$$\delta^{ARC}_N = \frac{2}{1 - \sgn((Q_i^\top g)^\top z^Q_*) \sqrt{1 + 4 \frac{\sigma \|z^Q_\|_Q^3}{|Q_i^\top g|} |z^Q_*|}}.$$  

and $z^Q_*$ is given by (7). Thus, by substituting $z_N$ in the formula $s^{ARC}_{N,i} = Q_i z_N$, one gets

$$s^{ARC}_{N,i} = Q_i \left( \frac{2}{1 - \sgn((Q_i^\top g)^\top z^Q_*) \sqrt{1 + 4 \frac{\sigma \|z^Q_\|_Q^3}{|Q_i^\top g|} |z^Q_*|}} \right) z^Q_* = \frac{2}{1 - \sgn(g^\top Q_i z^Q_*) \sqrt{1 + 4 \frac{\sigma \|z^Q_*\|^3}{|g| Q_i z^Q_*} |z^Q_*|}} Q_i z_N = \frac{2}{1 - \sgn(g^\top s^Q_i) \sqrt{1 + 4 \frac{\sigma \|s^Q_i\|^3}{|g| s^Q_i} |s^Q_i|}} s^Q_i.$$  

8
As in the exact case, the key assumption to ensure that, the step $s^{ARC}_{N,i}$ is a stationary point of the ARC subproblem, is the use of the M-norm where $M$ is satisfying the following assumption:

**Assumption 3.4** There exist iteration dependent parameters $\theta > 0$ and matrices $M$, such that $M$ is SPD and satisfies $Ms_i^Q = \frac{\theta}{g^T s_i^Q} g$, where $s_i^Q$ is approximate solution (in the subspace $L_i$) of the linear system $Bs = -g$.

**Theorem 3.5** Let Assumption 3.3 holds. The step $s^{ARC}_{N,i}$ is a stationary point for the subproblem (3) if and only if $M$ satisfies Assumption 3.4.

**Proof.** The proof is similar to the proof of Theorem 3.2 by considering $s_i^Q$ instead of $s_i^Q$.

Again, the existence of a matrix $M$ which satisfies Assumption 3.4 is not problematic as far as Assumption 3.3 holds. As explained before one may use BFGS techniques to find a matrix $M$.

Similarly to the case of exact computation, note that in the formula of $s^{ARC}_{N,i}$ the matrix $M$ is used only to compute the $M$-norm of $s_i^Q$. Therefore the explicit formula of the matrix $M$ is not needed, and only the value of $\theta = \|s_i^Q\|^2_M$ is only what is needed for the step computation. The following theorem gives a sufficient condition to ensure that $s^{ARC}_{N,i}$ is the unique minimizer of $m^C$ over the subspace $L_i$.

**Theorem 3.6** Let Assumptions 3.3 and 3.4 hold. If the matrix $Q_i^T (B + \sigma s^{ARC}_{N,i}M)Q_i$ is positive definite, then the step $s^{ARC}_{N,i}$ is the unique global minimizer of the subproblem (3) over the subspace $L_i$.

**Proof.** Indeed, using Assumption 3.4, one has $Ms_i^Q = \frac{\theta}{g^T s_i^Q} g$ meaning that $Q_i^T M Q_i z_i^Q = \frac{\theta}{(Q_i^T g)^T z_i^Q} g$. Hence, if we suppose that the matrix $Q_i^T (B + \lambda N M)Q_i$ is positive definite, by applying Theorem 3.3 to the reduced subproblem (13), we see that the step $z_N$ is the unique global minimizer of the subproblem (13). We conclude that $s^{ARC}_{N,i} = Q_i z_N$ is the global minimizer of the subproblem (3) over the subspace $L_i$.

## 4 An LS algorithm in the spirit of ARC

In this section, we derive the new LS algorithm on the base of ARC when the proposed $M$-scaled norm is used. For the sake of clarity, the subscript $k$ is reused to indicate the outer iteration index all over our analysis. In the sequel, $s_k^Q$ will denote the (approximate or exact) solution of the linear system $B_k s = -g_k$.

As far as Assumptions 3.1 and 3.2 (or 3.3 and 3.4 when subspace methods are used) hold, we note that unsuccessful iterations of Algorithm 1 require only updating the value of $\sigma_k$ and the current step direction is kept unchanged. In this case, the approximate solutions of the subproblem is obtained only by updating the step-size $\delta_k$. This means that the computational cost of unsuccessful iterations is getting cheap (see Theorems 3.1-3.4) compared to solving subproblem as required by the classical ARC when the Euclidean norm is used (see e.g. [5]). As a consequence, the use of the specific proposed $M_k$-norm in Algorithm 1 leads to a new algorithm where the dominant computational cost is the cost of successful iterations. With such choice of
the matrix $M_k$, the classical ARC algorithm behaves as an LS method with a specific backtracking strategy and an acceptance criteria in the spirit of the ARC algorithm. The trial step is of the form $s_k = \delta_k s^Q_k$ where $s^Q_k$ is the (approximate) solution of $B_k s = -g_k$, and the step length $\delta_k > 0$ is chosen such as

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{f(x_k) - m^Q_k(s_k)} \geq \eta,$$

or, equivalently,

$$f(x_k + s_k) \leq f(x_k) + \eta \left(s^\top_k g_k + \frac{1}{2}s^\top_k B_k s_k\right). \quad (15)$$

The step length $\delta_k$ is computed as follows:

$$\delta_k = \frac{2}{1 - \text{sgn}(g_k^\top s^Q_k)\sqrt{1 + 4\frac{\sigma_k\theta_k^2}{s_k^Q g_k}}}, \quad (16)$$

where $\theta_k > 0$ and $\sigma_k$ is initially equals to the current value of the regularization parameter (as in the original ARC algorithm). For large values of $\delta_k$ the sufficient decrease condition (15) may not be satisfied. In this case, the value of $\sigma_k$ is enlarged using an expansion factor $\nu_\sigma > 1$. Iteratively, the value of $\delta_k$ is updated (when $\sigma_k$ increases, the absolute value of $\delta_k$ decreases) and the acceptance condition (15) is checked again, until the decrease condition in (15) is satisfied. We denote the modified ARC algorithm by LS-ARC. To extend the convergence and complexity analysis of the classical ARC to the proposed LS-ARC algorithm one need to show that the $M_k$-scaled norms is uniformly equivalent to the Euclidean norm. Under appropriate assumptions, the next theorem shows that there exists a nonempty subset of choices for the parameter $\theta_k$ such as the $M_k$-scaled norm is uniformly equivalent to the Euclidean one.

**Theorem 4.1** There exists a nonempty closed interval $I_k \subset \mathbb{R}^+_+$ such that for all $\theta_k \in I_k$ the scaled $M_k$-norms are uniformly equivalent to the $l_2$-norm on $\mathbb{R}^n$.

**Proof.** Let $s^Q_k = \frac{s^Q_k}{\|s^Q_k\|}$ and $\bar{g}_k$ be an orthonormal vector to $s^Q_k$ (i.e., $\|\bar{g}_k\| = 1$ and $\bar{g}_k^\top s^Q_k = 0$) such that

$$\frac{g_k}{\|g_k\|} = \cos(\alpha_k) s^Q_k + \sin(\alpha_k) \bar{g}_k, \quad \text{where } \cos(\alpha_k) = \frac{g_k^\top s^Q_k}{\|g_k\|\|s^Q_k\|}.$$

Using Assumption 3.4, One has

$$M_k s^Q_k = \frac{\theta_k g_k}{g_k^\top s^Q_k \|s^Q_k\|} = \frac{\theta_k \|g_k\|}{g_k^\top s^Q_k \|s^Q_k\|} \left(\cos(\alpha_k) s^Q_k + \sin(\alpha_k) \bar{g}_k\right) = \frac{\theta_k}{\|s^Q_k\|^2} s^Q_k + \frac{\theta_k \tan(\alpha_k)}{\|s^Q_k\|^2} \bar{g}_k.$$

Moreover $M_k$ is symmetric, hence there exists a parameter $\gamma_k$ such that

$$M_k = \begin{bmatrix} s^Q_k & \bar{g}_k \end{bmatrix} N_k \begin{bmatrix} s^Q_k & \bar{g}_k \end{bmatrix}^\top \text{ where } N_k = \begin{bmatrix} \frac{\theta_k}{\|s^Q_k\|^2} & \frac{\theta_k \tan(\alpha_k)}{\|s^Q_k\|^2} \\ \frac{\theta_k \tan(\alpha_k)}{\|s^Q_k\|^2} & \gamma_k \end{bmatrix}. $$
Choosing a uniformly bounded matrix $N_k$ is then equivalent to impose the appropriate conditions on $\theta_k$ and $\gamma_k$. In the sequel of the proof, such conditions will be outlined.

The eigenvalues $\lambda_k^{\text{min}}$ and $\lambda_k^{\text{max}}$ of the matrix $N_k$ are the roots of

$$
\lambda^2 - \left( \frac{\theta_k}{\|s^Q_k\|^2} + \gamma_k \right) \lambda + \frac{\theta_k}{\|s^Q_k\|^2} \gamma_k - \left( \frac{\theta_k}{\|s^Q_k\|^2} \tan(\alpha_k) \right)^2 = 0,
$$
hence

$$
\lambda_k^{\text{min}} = \frac{\left( \frac{\theta_k}{\|s^Q_k\|^2} \tan(\alpha_k) \right)^2 + \lambda_k^{\text{min}} \left( \frac{\theta_k}{\|s^Q_k\|^2} - \lambda_k^{\text{min}} \right)}{2} \quad \text{and} \quad \lambda_k^{\text{max}} = \frac{\left( \frac{\theta_k}{\|s^Q_k\|^2} + \gamma_k \right)^2 + \sqrt{\omega_k}}{2},
$$

where $\omega_k = \left( \frac{\theta_k}{\|s^Q_k\|^2} - \gamma_k \right)^2 + 4 \left( \frac{\theta_k}{\|s^Q_k\|^2} \tan(\alpha_k) \right)^2$. Note that both eigenvalues are monotonically increasing as functions of $\gamma_k$.

Assuming that $\frac{\theta_k}{\|s^Q_k\|^2} > \lambda_k^{\text{min}}$, one can deduce from the expression of $\lambda_k^{\text{min}}$ the following:

$$
\gamma_k = \frac{\left( \frac{\theta_k}{\|s^Q_k\|^2} \tan(\alpha_k) \right)^2 + \lambda_k^{\text{min}} \left( \frac{\theta_k}{\|s^Q_k\|^2} - \lambda_k^{\text{min}} \right)}{2}.
\tag{17}
$$

From Assumption 3.3, i.e., $|g_k^\top s^Q_k| \geq \epsilon_d \|g_k\| \|s^Q_k\|$ where $\epsilon_d > 0$, one has $\tan(\alpha_k)^2 \leq \tau_d \triangleq \frac{1 - \epsilon_d^2}{\epsilon_d^2}$. Hence,

$$
\gamma_k \leq \frac{\left( \frac{\theta_k}{\|s^Q_k\|^2} \right)^2 + \lambda_k^{\text{min}} \left( \frac{\theta_k}{\|s^Q_k\|^2} - \lambda_k^{\text{min}} \right)}{2} \tau_d + \frac{\theta_k}{\|s^Q_k\|^2} \left( \frac{\theta_k}{\|s^Q_k\|^2} - \lambda_k^{\text{min}} \right)
$$

One may choose $\theta_k$ of the form $\theta_k = \beta_k \|s^Q_k\|^2$ where $\beta_k \in (\beta_{\text{min}}, \beta_{\text{max}})$ with $\beta_{\text{min}} < \beta_{\text{max}}$ are positive scalars. In this case, $\gamma_k$ is uniformly bounded from above, and by monotonicity, $\lambda_k^{\text{max}}$ is also uniformly bounded from above.

To sum up, under Assumptions 3.3 and 3.4, choosing $\theta_k$ of the form $\theta_k = \beta_k \|s^Q_k\|^2$ and $\lambda_k^{\text{min}} \in [\beta_{\text{min}}, \beta_k)$ leads to a uniformly bounded matrix $N_k$ with the eigenvalues $\lambda_k^{\text{min}}$ and $\lambda_k^{\text{max}} = \beta_k + \gamma_k - \lambda_k^{\text{min}}$ where $\gamma_k$ is given by (17).

A possible choice for the matrix $M_k$ can be obtained by completing the vectors family $\{\bar{s}^Q_k, \bar{g}_k\}$ to an orthonormal basis $\{s^Q_k, \bar{g}_k, q_3, q_4, \ldots, q_n\}$ of $\mathbb{R}^n$ as follows:

$$
M_k = [s^Q_k, \bar{g}_k, q_3, \ldots, q_n] \begin{bmatrix} N_k & 0 \\ 0 & D \end{bmatrix} [s^Q_k, \bar{g}_k, q_3, \ldots, q_n]^\top,
$$

where $D = \text{diag}(d_3, \ldots, d_n) \in \mathbb{R}^{(n-2) \times (n-2)}$ with positive diagonal entrees independent from $k$. One concludes that there exists a nonempty closed interval $I_k \subset \mathbb{R}^+$ such that

$$
\{\theta_k = \beta_k \|s^Q_k\|^2 \quad \text{where} \quad \beta_k \in (\beta_{\text{min}}, \beta_{\text{max}})\} \subset I_k,
\tag{18}
$$

and for all $\theta_k \in I_k$ the scaled $M_k$-norm is uniformly equivalent to the $l_2$-norm on $\mathbb{R}^n$. ■
In the next lemma, we show that when our proposed $M_k$-norm is used, the condition (6) imposed on the cubic model $m_k^C$ can be expressed only in terms of $s_k^Q$ and $\nabla m_k^Q$. The latter condition is required in the algorithm at each iteration to ensure that it takes at most $O(\epsilon^{-3/2})$ function-evaluation (and iteration) to reduce the gradient norm below $\epsilon \in (0,1)$.

**Lemma 4.1** Let Assumptions 3.3 and 3.4 hold. Then imposing the condition (6) is equivalent to the following condition

$$
\|\nabla m_k^Q(s_k^Q)\| \leq \frac{2 \text{sgn}(g_k^T s_k^Q) \zeta}{1 + \text{sgn}(g_k^T s_k^Q) \sqrt{1 + 4 \sigma g_k^T s_k^Q / |g_k s_k^Q|}} \|s_k^Q\|^2.
$$

**Proof.** In the following proof we choose to omit the outer iteration index $k$. One has $s = \alpha_N d_N = -\alpha_N s_k^Q$ where $\alpha_N > 0$ satisfies (10). Hence,

$$
\nabla m_k^C(s) = g + Bs + \sigma \|s\|_M Ms = g - \frac{\alpha_N}{g^T s_k^Q} Bs^Q - \frac{\sigma \alpha_N^2 \|s\|_M^2}{|g^T s_k^Q|^3} M s_k^Q - \frac{\sigma \alpha_N^2 \|s\|_M^2}{|g^T s_k^Q|^3} s_k^Q.
$$

Since the matrix $M$ satisfies $Ms^Q = \frac{\theta}{g^T s_k^Q} g$ where $\theta = \|s^Q\|_M^3$, one gets

$$
\nabla m_k^C(s) = g - \frac{\alpha_N}{g^T s_k^Q} Bs^Q - \sigma \alpha_N^2 \|s\|_M^2 M s_k^Q = \left(1 - \frac{\sigma \alpha_N^2 \|s\|_M^2}{|g^T s_k^Q|^3}\right) g - \frac{\alpha_N}{g^T s_k^Q} Bs^Q.
$$

By definition of $\alpha_N$ (see (10)), one has $\left(1 - \frac{\sigma \alpha_N^2 \|s\|_M^2}{|g^T s_k^Q|^3}\right) = -\frac{\alpha_N}{g^T s_k^Q}$, which means

$$
\nabla m_k^C(s) = -\frac{\alpha_N}{g^T s_k^Q} (g + Bs^Q) = -\frac{\alpha_N}{g^T s_k^Q} \nabla m_k^Q(s^Q).
$$

Hence, the condition (6) is being equivalent to

$$
\|\nabla m_k^Q(s_k^Q)\| \leq \frac{\zeta |g^T s_k^Q|}{\alpha_N} \|s\|^2 = \frac{2 \text{sgn}(g_k^T s_k^Q) \zeta}{1 + \text{sgn}(g_k^T s_k^Q) \sqrt{1 + 4 \sigma g_k^T s_k^Q / |g_k s_k^Q|}} \|s_k^Q\|^2.
$$

Algorithm 2 details, using Lemma 4.1, the adaptation of the classical algorithm ARC when our proposed $M_k$-norm is used.

In Step 2 of Algorithm 2, the use of an exact solver to compute $s_k^Q$ implies that $\|\nabla m_k^Q(s_k^Q)\| = 0$ for all iterations. Thus the condition (19) is automatically satisfied for each iteration and the “Go to” instruction in the backtracking strategy will be avoided. When a subspace method is used to approximate the step $s_k^Q$, we note the important freedom to add an additional preconditioner to the problem. In this case, one would solve the quadratic problem with preconditioning until the criterion (19) is met. This is expected to happen early along the Krylov iterations when the preconditioner for the linear system $B_k s = -g_k$ is good enough.

As mentioned earlier, for each outer iteration $k$ in Algorithm 2, the matrix $M_k$ is not used explicitly in the computations. In fact, only the value of the parameter $\theta_k = \|s_k^Q\|_M > 0$ is required in our proposed algorithm. One possible choice for $\theta_k$ is $|g_k^T s_k^Q|$, such choice is equivalent, when the model Hessian matrix is SPD, to work with the energy norm in the ARC
Algorithm 2: The LS-ARC algorithm.

Data: select an initial point $x_0$, the constant $0 < \eta < 1$ and $\zeta > 0$. Set the initial regularization $\sigma_0 > 0$, and the constants $0 < \nu_1 \leq 1 < \nu_2$.

for $k = 1, 2, \ldots$ do

Step 1: compute the model gradient $g_k$ and an approximated Hessian $B_k$;

Step 2: choose a parameter $\theta_k$ (in the sense of (18)) and a step $s_k^Q$ such as:

$$
\|\nabla m_k^Q(s_k^Q)\| \leq \frac{2 \text{sgn}(g_k^T s_k^Q) \zeta}{-1 + \text{sgn}(g_k^T s_k^Q) \sqrt{1 + 4 \frac{\sigma_k \theta_k^{3/2}}{|g_k^T s_k^Q|}} s_k^Q \|s_k^Q\|};
$$

Step 3: set $\delta_k \leftarrow 2 \left(1 - \text{sgn}(g_k^T s_k^Q) \sqrt{1 + 4 \frac{\sigma_k \theta_k^{3/2}}{|g_k^T s_k^Q|}} \right)^{-1}$;

Step 4: repeat

set $\sigma_k \leftarrow \nu_2 \sigma_k$ and $\delta_k \leftarrow 2 \left(1 - \text{sgn}(g_k^T s_k^Q) \sqrt{1 + 4 \frac{\sigma_k \theta_k^{3/2}}{|g_k^T s_k^Q|}} \right)^{-1}$;

if $\|\nabla m_k^Q(s_k^Q)\| > \zeta |\delta_k| \|s_k^Q\|^2$ then

| Go to Step 2;

end

until $f(x_k + \delta_k s_k^Q) \leq f(x_k) + \eta \left(\delta_k |s_k^Q|^T g_k + \frac{s_k^Q}{2} |s_k^Q|^T B_k |s_k^Q|\right)$;

Step 5: set $s_k = \delta_k s_k^Q$, $x_{k+1} = x_k + s_k$ and $\sigma_{k+1} = \nu_1 \sigma_k$;

end

algorithm (see [1]). To encourage the Algorithm to take (when it is possible) the Newton step other possible choices for $\theta_k$ can be considered (see Section 6).

As far as $\theta_k \in I_k$, the proposed $M_k$-norm will be uniformly equivalent to the $l_2$ one along the iterations, all the convergence results of the classical ARC algorithm [5, 4, 2] apply as well for Algorithm 2. In particular, the LS-ARC algorithm is guaranteed to have an improved function-evaluation worst-case complexity of order $\epsilon^{-3/2}$ to ensure (5) as stated by the next theorem.

Theorem 4.2 Let Assumption 3.3 holds. Then under regularity assumptions on $f$ (as in [2, 4]), given an $\epsilon > 0$, Algorithm 2 needs at most

$$
\left\lceil \frac{f(x_0) - f_{\text{low}}}{\epsilon^{3/2}} \right\rceil
$$

iterations and at most

$$
\left\lceil \frac{f(x_0) - f_{\text{low}}}{\epsilon^{3/2}} \right\rceil \left(1 + \frac{\log(\nu_1)}{\log(\nu_2)} + \frac{1}{\log(\nu_2)} \log \left(\frac{\sigma_0}{\sigma_{\text{max}}}\right) \right)
$$

evaluation of $f$ to produce an iterate $x_\epsilon$ such that $\|\nabla f(x_\epsilon)\| \leq \epsilon$ where $f_{\text{low}}$ is a lower bound on $f$, $\sigma_{\text{max}}$ and $\kappa_s$ are positive constants.

Proof. Using Theorem 4.1 and Lemma 4.1, the proof is the same as [2, Theorem 2.5].
5 An LS algorithm in the spirit of TR

Similarly to ARC algorithm, it is possible to render TR algorithm behaves as an LS algorithm using the same scaled norm to define the trust-region neighborhood. As a reminder in a basic TR algorithm [8], one computes a trial step $s^{TR}_k$ by approximately solving

$$\min_{s \in \mathbb{R}^n} m^Q_k(s) \quad \text{s. t.} \quad \|s\|_{M_k} \leq \Delta_k,$$  \hspace{1cm} (20)

where $\Delta_k > 0$ is known as the TR radius. As in ARC algorithms, the scaled norm $\|\cdot\|_{M_k}$ may vary along the iterations and $M_k$ is an SPD matrix.

Once the trial step $s^{TR}_k$ is determined, the objective function is computed at $x_k + s^{TR}_k$ and compared with the value predicted by the model at this point. If the model value predicts sufficiently well the objective function (i.e., the iteration is successful), the trial point $x_k + s^{TR}_k$ will be accepted and the TR radius is eventually expanded (i.e., $\Delta_{k+1} = \tau_2 \Delta_k$ with $\tau_2 \geq 1$). If the model turns out to predict poorly the objective function (i.e., the iteration is unsuccessful), the trial point is rejected and the TR radius is contracted (i.e., $\Delta_{k+1} = \tau_1 \Delta_k$ with $\tau_1 < 1$). The ratio between the actual reduction and the predicted reduction for the TR algorithms is defined as in ARC (see (4)). For a given scalar $0 < \eta < 1$, the iteration will be said successful if $\rho \geq \eta$, and unsuccessful otherwise. Algorithm 3 gives a detailed description of a basic TR algorithm.

**Algorithm 3: TR algorithm.**

**Data:** select an initial point $x_0$ and $0 < \eta < 1$. Set the initial TR radius $\Delta_0 > 0$ and the constants $0 \leq \tau_1 < 1 \leq \tau_2$.

for $k = 1, 2, \ldots$ do

Step 1: compute the model gradient $g_k$ and an approximated Hessian $B_k$;

Step 2: compute the step $s^{TR}_k$ as an approximate solution of (20);

Step 3: if $\rho_k \geq \eta$ then

| set $x_{k+1} = x_k + s^{TR}_k$ and $\Delta_{k+1} = \tau_2 \Delta_k$ |

else

| set $x_{k+1} = x_k$ and $\Delta_{k+1} = \tau_1 \Delta_k$ |

end

end

We will focus on the TR subproblem minimization for a given iteration $k$. The outer iteration subscript $k$ will be dropped to keep the notations simple. As for ARC algorithm, we will call the Newton-like step associated with the TR subproblem the vector of the following form

$$s^N = \alpha_N d_N, \quad \text{where} \quad \alpha_N = \arg \min_{t > 0: \|td_N\|_M \leq \Delta} m^Q(td_N).$$  \hspace{1cm} (21)

Note that for the TR subproblem, the solution we are looking for lies either interior to the trust region, that is $\|s\|_M < \Delta$, or on the boundary, $\|s\|_M = \Delta$. If the solution is interior, the solution $s^{TR}$ is the unconstrained minimizer of the quadratic model $m^Q$. Such scenario can only happen if $m^Q$ is convex, that is, the Hessian matrix $B$ is SPD. In the non convex case a solution lies on the boundary of the trust region, while in the convex case a solution may or may not do so. Consequently in practice, the TR algorithm finds first the unconstrained minimizer of the model
If the model is unbounded from below, or if the unconstrained minimizer lies outside the trust region, the minimizer then occurs on the boundary of the trust region. Similarly to ARC algorithm, one has the following results:

**Theorem 5.1** Let Assumption 3.1 holds.

1. The Newton-like step (21) is of the following form:

   \[ s_{TR}^N = \delta_{TR}^N s^*_Q, \text{ where } \delta_{TR}^N = \min \left( 1, -\text{sgn}(g^\top s^*_Q) \frac{\Delta}{\|s^*_Q\|_M} \right). \]

2. The step \( s_{TR}^N \) (when it lies on the border of the trust region) is a stationary point of the subproblem (20) if and only if \( Ms^Q = \theta g^\top s^*_Q \) where \( \theta = \|s^*_Q\|_M^2 \).

3. Moreover, when Assumption 3.2 holds and \( s_{TR}^N \) lies on the border, if the matrix \( B + \lambda_N^TR M \) is positive definite, then \( s_{TR}^N \) is the unique global minimizer of the subproblem (20) where \( \lambda_N^TR = g^\top s^*_Q \theta \left( 1 + \text{sgn}(g^\top s^*_Q) \frac{\Delta}{\|s^*_Q\|_M} \right) \).

**Proof.**

1. To calculate the Newton step \( s_{TR}^N \), we first note, for all \( t > 0 \)

   \[ m_Q(td_N) = m_Q(0) - t - \frac{1}{g^\top s^*_Q} \frac{t^2}{2}. \]  

   Consider the case where the curvature model along the Newton direction is positive, that is when \( g^\top s^*_Q < 0 \), and compute the value of the parameter \( t \) at which the unique minimizer of (22) is attained. Let \( t^* \) denotes this optimal parameter. Taking the derivative of (22) with respect to \( t \) and equating the result to zero, one has \( t^* = -g^\top s^*_Q \). Two sub-cases may then occur. The first is when this minimizer lies within the trust region (i.e. \( t^*\|d_N\|_M \leq \Delta \)), then

   \[ \alpha_N = t^* = -g^\top s^*_Q. \]

   If on the other hand, \( t^*\|d_N\|_M > \Delta \), then the line minimizer is outside the trust region and we have that

   \[ \alpha_N = \frac{\Delta}{\|d_N\|_M} = -g^\top s^*_Q \frac{\Delta}{\|s^*_Q\|_M}. \]

   Finally, we consider the case where the curvature of the model along the Newton-like step is negative, that is, when \( g^\top s^*_Q > 0 \). In that case, the minimizer lies on the boundary of the trust region, and thus

   \[ \alpha_N = \frac{\Delta}{\|d_N\|_M} = g^\top s^*_Q \frac{\Delta}{\|s^*_Q\|_M}. \]

   By combining all cases, one concludes that

   \[ s_{TR}^N = \delta_{TR}^N s^*_Q, \text{ where } \delta_{TR}^N = \min \left( 1, -\text{sgn}(g^\top s^*_Q) \frac{\Delta}{\|s^*_Q\|_M} \right). \]
2. Suppose that the Newton-like step lies on the border of the trust region i.e., $s_{TRN}^N = \delta_{TRN} s^Q = -\text{sgn}(g^\top s^Q) \frac{\lambda}{\|s^Q\|_M} s^Q$. The latter step is a stationary point of the subproblem (20) if and only if there exists a Lagrange multiplier $\lambda \geq 0$ such that 

$$(B + \lambda M)s_{TRN}^N = -g.$$

Substituting $s_{TRN}^N = \delta_{TRN} s^Q$ in the latter equation, one has

$$\lambda Ms^Q = \left(1 - \frac{1}{\delta_{TRN}}\right) g.$$

By multiplying it from left by $(s^Q)^\top$, we deduce that

$$\lambda = \left(1 - \frac{1}{\delta_{TRN}}\right) \frac{g^\top s^Q}{\|s^Q\|_M^2} = \frac{g^\top s^Q}{\theta} \left(1 + \text{sgn}(g^\top s^Q) \frac{\|s^Q\|_M}{\Delta}\right).$$

By replacing the value of $\lambda$ in (23), we obtain that $Ms^Q = \frac{\theta}{g^\top s^Q} g$ where $\theta = \|s^Q\|_M > 0$.

3. Indeed, If the step $s_{TRN}^N$ lies on the boundary of the trust-region (i.e. $\|s_{TRN}^N\|_M = \Delta$), One has $Ms^Q = \frac{\theta}{g^\top s^Q} g$. Then applying (2) of Theorem 5.1, we see that

$$(B + \lambda_{TRN}^N M)s_{TRN}^N = -g$$

with $\lambda_{TRN}^N = \frac{g^\top s^Q}{\theta} \left(1 + \text{sgn}(g^\top s^Q) \frac{\|s^Q\|_M}{\Delta}\right) > 0$. Applying [8, Theorem 7.4.1], we see that if we assume the matrix $B + \lambda_{TRN}^N M$ to be positive definite, then $s_{TRN}^N$ is the unique global minimizer of the subproblem (20).

Given an SPD matrix $M$ that satisfies the secant equation $Ms^Q = \frac{\theta}{g^\top s^Q} g$, item (3) of Theorem 5.1 states that the step $s_{TRN}^N$ is the global minimizer of the associated subproblems as far as the matrix $B + \lambda_{TRN}^N M$ is SPD. We note that $\lambda_{TRN}^N$ goes to infinity as the trust region radius $\Delta$ goes to zero, meaning that the matrix $B + \lambda_{TRN}^N M$ will be SPD as far as $\Delta$ is chosen to be sufficiently small. Since the TR update mechanism allows to shrink the value of $\Delta$ (when the iteration is declared as unsuccessful), satisfying the targeted condition will be geared automatically by the TR algorithm.

In the case where the linear system $Bs = -g$ is solved iteratively (by projection onto a sequence embedded subspaces). Then using similar arguments as for ARC algorithm (as far as Assumptions 3.3 and 3.4 hold), one can extend the results in Theorem 5.1 when a truncated Newton-like step is used in the TR algorithm.

Again, we note that unsuccessful iterations in the TR Algorithm require only updating the value of the TR radius $\Delta$ and the current step direction is kept unchanged. For such iterations, as far as the matrix $M$ satisfies Assumption 3.4, the approximate solution of the TR subproblem is obtained only by updating the step-size $\delta$. This means that the computational cost of unsuccessful iterations do not requires solving any extra subproblem. As a consequence, we obtain a new algorithm that behaves as an LS method with a specific backtracking strategy and an acceptance criteria in the sprite of the TR algorithm. For the $k^{th}$ iteration, the trial step is of the form $s_k = \delta_k s^Q_k$ where $s^Q_k$ is the (approximate) solution of the linear system $B_k s = -g_k$ and the step length $\delta_k > 0$ is chosen such as

$$f(x_k + s_k) \leq f(x_k) + \eta \left(s^\top_k g_k + \frac{1}{2} s^\top_k B_k s_k\right).$$

(23)
The appropriate value of $\delta_k$ is selected as

$$\delta_k = \min \left( 1, -\text{sgn}(g_k^T s_k^Q) \frac{\Delta_k}{\theta_k^{1/2}} \right), \quad (24)$$

where $\theta_k > 0$ and $\Delta_k$ is initially equals to the current value of the TR radius (as in the original TR algorithm). For large values of $\delta_k$ the sufficient decrease condition (23) may not be satisfied, in this case, the value of $\Delta_k$ is contracted using a contraction factor $\tau_1$. Iteratively, the value of $\delta$ is updated and the acceptance condition (23) is checked again, until the sufficient decrease condition is satisfied. Algorithm 4 details the adaptation of the classical TR algorithm when our proposed $M_k$-norm is used. We denote the modified TR algorithm by LS-TR.

**Algorithm 4: The LS-TR algorithm.**

**Data:** select an initial point $x_0$ and $0 < \eta < 1$. Set the initial TR radius $\Delta_0 > 0$ and the constants $0 \leq \tau_1 < 1 \leq \tau_2$.

**for** $k = 1, 2, \ldots$ **do**

**Step 1:** compute the model gradient $g_k$ and an approximated Hessian $B_k$;

**Step 2:** choose a parameter $\theta_k$ (in the sense of (18)) and an approximate stationary point $s_k^Q$ of $m_k^Q$;

**Step 3:** set $\delta_k \leftarrow \min \left( 1, -\text{sgn}(g_k^T s_k^Q) \frac{\Delta_k}{\theta_k^{1/2}} \right)$;

**Step 4:** repeat

set $\Delta_k \leftarrow \tau_1 \Delta_k$ and $\delta_k \leftarrow \min \left( 1, -\text{sgn}(g_k^T s_k^Q) \frac{\Delta_k}{\theta_k^{1/2}} \right)$;

until $f(x_k + \delta_k s_k^Q) \leq f(x_k) + \eta \left( \delta_k [s_k^Q]^T g_k + \frac{\delta_k^2}{2} [s_k^Q]^T B_k [s_k^Q] \right)$;

**Step 5:** set $s_k = \delta_k s_k^Q$, $x_{k+1} = x_k + s_k$ and $\Delta_{k+1} = \tau_2 \Delta_k$;

**end**

As LS-ARC algorithm, for each outer iteration $k$ in Algorithm 4, the matrix $M_k$ is not used explicitly in the computations but only through to the value of $\theta_k = \|s_k^Q\|_M^2 > 0$. To extend the same convergence and complexity results of the classical TR algorithm, the choice of the parameter $\theta_k$ has to satisfy the requirement that makes the proposed $M_k$-norm will be uniformly equivalent to the $l_2$ one along the iterations (see Theorem 4.1). For the sake of obtaining good practical performance, one may need to choose $\theta_k$ (in addition to latter requirement) so that it encourages the LS-TR algorithm to take the Newton step along the outer iterations (see Section 6).

**6 Numerical experiments**

In this section we report the results of experiments performed in order to assess the efficiency and the robustness of the proposed algorithms (LS-ARC and LS-TR) compared with the classical LS algorithm using the standard Armijo rule. In the latter approach, the trial step is of the form $s_k = \delta_k \text{sgn}(-g_k^T s_k^Q) s_k^Q$ where $s_k^Q$ is a given approximate stationary point of $m_k^Q$ such as $g^T s_k^Q \neq 0$, and the step length $\delta_k > 0$ is chosen such as

$$f(x_k + s_k) \leq f(x_k) + \eta s_k^T g_k, \quad (25)$$
where $\eta \in (0, 1)$. The appropriate value of $\delta_k$ is estimated using a backtracking approach with a contraction factor set to $\tau \in (0, 1)$ and where the step length is initially chosen to be 1. This LS method will be called LS-ARMIJO. We implement all the the algorithms as Matlab m-files and for all the tested algorithms $B_k$ is set to the true Hessian $\nabla^2 f(x_k)$.

By way of comparison, we have also implemented the standard ARC and TR algorithms (see Algorithms 1 and 3) using the Lanczos-based solver GLTR/GLRT implemented in GALAHAD [12]. The two subproblem solvers, GLTR/GLRT are implemented in Fortran and interfaced with Matlab using the default parameters. For the subproblem formulation we used the $l_2$ norm (i.e., for all iterations the matrix $M_k$ is set to identity). We shall refer to the ARC/TR methods based on GLRT/GLTR as GLRT-ARC/GLTR-TR.

The other parameters defining the implemented algorithms are set as follows, for GLRT-ARC and LS-ARC

$$\eta = 0.1, \ \nu_1 = 0.5, \ \nu_2 = 2, \ \text{and} \ \sigma_0 = 1;$$

for GLTR-TR and LS-TR

$$\eta = 0.1, \ \tau_1 = 0.5, \ \tau_2 = 2, \ \text{and} \ \Delta_0 = 1;$$

and last for LS-ARMIJO

$$\eta = 0.1, \ \text{and} \ \tau = 0.5.$$

In all algorithms the maximum number of iterations is set to 10000 and the algorithms stop when

$$\|g_k\| \leq 10^{-5}.$$

A crucial ingredient in LS-ARC and LS-TR is the management of the parameter $\theta_k$. A possible choice for $\theta_k$ is $|g_k^\top s_k^Q|$. This choice is inspired from the fact that, when the Hessian matrix $B_k$ is SPD, this update corresponds to use the energy norm (i.e., set the matrix $M_k$ equals to $B_k$) (see [1] for more details). However, this choice did not lead to good performance of the algorithms LS-ARC and LS-TR. In our implementation, for the LS-ARC algorithm, we set the value of $\theta_k$ as follows:

$$\theta_k = \beta_k \|s_k^Q\|^2,$$

where $\beta_k = \kappa \sigma_k^{-2/3}$ and $\kappa = 10^{-4}$ if $g_k^\top s_k^Q < 0$ and 2 otherwise. By this choice, we are willing to allow LS-ARC, at the start of the backtracking procedure, to take a step that is (approximately) the Newton step. Similarly, for the LS-TR method, we set $\theta_k = \|s_k^Q\|^2$ in fact as such value is of the same order as $\Delta_k^2$ this allows LS-TR to use the Newton step at the start of the backtracking strategy (as in LS-ARMIJO method). Within this choice of $\theta_k$, we ensure that the $M_k$-scaled norm will be uniformly equivalent to the $l_2$-norm for all iterations (see Theorem 4.1).

All the Algorithms are evaluated on a set of unconstrained optimization problems from the CUTEst collection [13]. The test set contains 62 large-scale ($1000 \leq n \leq 10000$) CUTest test problem with their default parameters. Regarding the algorithms LS-TR, LS-ARC, and LS-ARMIJO, we approximate the solution of the subproblem (7) using the MINRES Matlab solver. The latter method is a Krylov subspace method designed to solve symmetric linear systems [17]. We run the algorithms with the MINRES default parameters except the relative tolerance error which is set to $10^{-4}$. We note that on the tested problems, the LS algorithms satisfied Assumption 3.3 (with $\epsilon_d = 10^{-12}$) for all outer iterations.

To compare the performance of the algorithms we use performance profiles proposed by Dolan and Moré [11] over a variety of problems. Given a set of problems $\mathcal{P}$ (of cardinality $|\mathcal{P}|$)
and a set of solvers $S$, the performance profile $\rho_s(\tau)$ of a solver $s$ is defined as the fraction of problems where the performance ratio $r_{p,s}$ is at most $\tau$

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

The performance ratio $r_{p,s}$ is in turn defined by

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : s \in S\}},$$

where $t_{p,s} > 0$ measures the performance of the solver $s$ when solving problem $p$ (seen here as the function evaluation, the gradient evaluation, and the CPU time). Better performance of the solver $s$, relatively to the other solvers on the set of problems, is indicated by higher values of $\rho_s(\tau)$. In particular, efficiency is measured by $\rho_s(1)$ (the fraction of problems for which solver $s$ performs the best) and robustness is measured by $\rho_s(\tau)$ for $\tau$ sufficiently large (the fraction of problems solved by $s$). Following what is suggested in [11] for a better visualization, we will plot the performance profiles in a log$_2$-scale (for which $\tau = 1$ will correspond to $\tau = 0$).

We present the obtained performance profiles in Figure 1. Regarding the gradient evaluation (i.e., outer iteration) performance profile, see Figure 1(a), LS approaches are the most efficient among all the tested solvers (in more 60% of the tested problems LS methods perform the best, while GLRT-ARC and GLTR-TR are performing better only in less than 15%). When it comes to robustness, all the tested approaches exhibit good performance (GLRT-ARC and GLTR-TR are slightly better and LS-ARMijo displays the worst performance).

For function evaluation performance profile (Figure 1(b)), GLRT-ARC, GLTR-TR show a better efficiency but not as good as LS methods (in more than 50% of the tested problems LS methods perform the best while GLRT-ARC, GLTR-TR are better only in less than 35%). The robustness of the tested algorithms is the same as in the outer iteration performance profile.

In terms of the demanded computing time, see Figure 1(c), as one can expect, GLRT-ARC and GLTR-TR are turned to be very consuming compare to the LS approaches. In fact, unlike the LS methods where only an approximate solution of one linear system is needed, the GLRT/GLTR approaches may require (approximately) solving multiple linear systems in sequence.

7 Conclusion

In this paper we have investigated the use of a specific scaled norm to define the cubic regularization term in the ARC subproblem. With this norm choice, we have shown that the trial step of ARC is getting collinear to the Newton like step. The obtained ARC algorithm behaves as an LS approach with a specific backtracking strategy. Under mild assumptions, the proposed scaled norm was shown to be uniformly equivalent to the Euclidean norm. The obtained LS algorithm enjoys the same convergence and complexity properties as ARC. Similarly to ARC algorithm and using the same scaled norm to define the TR neighborhood, the proposed approach was generalized to cover the TR framework as well. Our numerical experiments showed encouraging performance of the proposed LS algorithms.

A number of issues need further investigation, in particular the best choice and the impact of the parameter $\theta_k$ on the performance of the proposed LS approaches. We have also assumed that along the iterations the Newton step is a sufficient descent direction for the objective function. It would be interesting to generalize the proposed algorithms when this assumption is not satisfied.
Figure 1: Performance profiles for 62 large scale optimization problems (i.e., $1000 \leq n \leq 10000$).

References


