A Line-Search Algorithm Inspired by the Adaptive Cubic Regularization Framework and Complexity Analysis

E. Bergou * Y. Diouane† S. Gratton‡

May 29, 2018

Abstract

Adaptive regularized framework using cubics has emerged as an alternative to line-search and trust-region algorithms for smooth nonconvex optimization, with an optimal complexity amongst second-order methods. In this paper, we propose and analyze the use of an iteration dependent scaled norm in the adaptive regularized framework using cubics. Within such scaled norm, the obtained method behaves as a line-search algorithm along the quasi-Newton direction with a special backtracking strategy. Under appropriate assumptions, the new algorithm enjoys the same convergence and complexity properties as adaptive regularized algorithm using cubics. The complexity for finding an approximate first-order stationary point can be improved to be optimal whenever a second order version of the proposed algorithm is regarded. In a similar way, using the same scaled norm to define the trust-region neighborhood, we show that the trust-region algorithm behaves as a line-search algorithm. The good potential of the obtained algorithms is shown 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

An unconstrained nonlinear optimization problem considers the minimization of a scalar function known as the objective function. Classical iterative methods for solving the previous problem are trust-region (TR) [8, 20], line-search (LS) [10] and algorithms using cubic regularization. The latter class of algorithms has been first investigated by Griewank [15] and then by Nesterov and Polyak [18]. Recently, Cartis *et al* [5] proposed a generalization to an adaptive regularized framework using cubics (ARC).

The worst-case evaluation complexity of finding an ϵ-approximate first-order critical point using TR or LS methods is shown to be computed in at most $\mathcal{O}(\epsilon^{-2})$ objective function or gradient evaluations, where $\epsilon \in]0, 1[$ is a user-defined accuracy threshold on the gradient norm

*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).

[17, 14, 7]. Under appropriate assumptions, ARC takes at most $\mathcal{O}(\epsilon^{-3/2})$ objective function or gradient evaluations to reduce the gradient of the objective function norm below ϵ, 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].

More 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 $\mathcal{O}(\epsilon^{-3/2})$ can be achieved by mean of a specific variable-norm in a TR method [16] 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 (by outer iteration, we mean the sequence of the iterates generated by the algorithm), this makes the computational cost per iteration expensive.

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, in [1] the proposed LS algorithm derived from ARC enjoys the same convergence and complexity analysis properties as ARC, in particular the first-order complexity bound of $\mathcal{O}(\epsilon^{-3/2})$. In the complexity analysis of ARC method [4], it is required that the Hessian approximation has to approximate accurately enough the true Hessian [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. We propose to use, in the regularization term of the ARC cubic model, an iteration dependent scaled norm. In this case, ARC behaves as an LS algorithm with a worst-case evaluation complexity of finding an ϵ-approximate first-order critical point of $\mathcal{O}(\epsilon^{-2})$ function or gradient evaluations. Moreover, under appropriate assumptions, a second order version of the obtained LS algorithm is shown to have a worst-case complexity of $\mathcal{O}(\epsilon^{-3/2})$.

The use of a scaled norm was first introduced in [8, Section 7.7.1] for TR methods where it was suggested to use the absolute-value of the Hessian matrix in the scaled norm, such choice was described as "the ideal trust region" that reflects the proper scaling of the underlying problem. For a large scale indefinite Hessian matrix, computing its absolute-value is certainly a computationally expensive task as it requires a spectral decomposition. This means that for large scale optimization problems the use of the absolute-value based norm can be seen as out of reach. Our approach in this paper is different as it allows the use of subspace methods.

In fact, as far as the quasi-Newton direction is not orthogonal with the gradient of the objective function at the current iterate, the specific choice of the scaled norm renders the ARC subproblem solution collinear with the quasi-Newton direction. Using subspace methods, we also consider the large-scale setting when the matrix factorizations are not affordable, implying that only iterative methods for computing a trial step can be used. Compared to the classical ARC, when using the Euclidean norm, the dominant computational cost regardless the function evaluation cost of the resulting algorithm is mainly the cost of solving a linear system for successful iterations. Moreover, the cost of the subproblem solution for unsuccessful iterations is getting inexpensive and requires only an update of a scalar. Hence, ARC behaves as an LS algorithm along the quasi-Newton direction, with a special backtracking strategy and an acceptance criteria in the sprite of ARC algorithm.

In this context, the obtained LS algorithm is globally convergent and requires a number of iterations of order ϵ^{-2} to produce an ϵ-approximate first-order critical point. A second order version of the algorithm is also proposed, by making use of the exact Hessian or at least of a good approximation of the exact Hessian, to ensure an optimal worst-case complexity bound of order $\epsilon^{-3/2}$. In this case, we investigate how the complexity bound depends on the quality of the chosen quasi-Newton direction in terms of being a sufficient descent direction. In fact, the obtained complexity bound can be worse than it seems to be whenever the quasi-Newton direction is approximately orthogonal with the gradient of the objective function. Similarly to ARC, we show that the TR method behaves also as an LS algorithm using the same scaled norm as in ARC. Numerical illustrations over a test set of large scale optimization problems are given in order to assess the efficiency of the obtained LS algorithms.

The proposed analysis in this paper assumes that the quasi-Newton direction is not orthogonal with the gradient of the objective function during the minimization process. When such assumption is violated, one can either modify the Hessian approximation using regularization techniques or, when a second order version of the LS algorithm is regarded, switch to the classical ARC algorithm using the Euclidean norm until this assumption holds. In the latter scenario, we propose to check first if there exists an approximate quasi-Newton direction, among all the iterates generated using a subspace method, which is not orthogonal with the gradient and that satisfies the desired properties. If not, one minimizes the model using the Euclidean norm until a new successful outer iteration is found.

We organize this paper as follows. In Section 2, we introduce the ARC method using a general scaled norm and derive the obtained LS algorithm on the base of ARC when a specific scaled norm is used. 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 discusses first how the iteration dependent can be chosen uniformly equivalent to the Euclidean norm, and then we propose a second order LS algorithm that enjoys the optimal complexity bound. The section ends with a detailed complexity analysis of the obtained algorithm. 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.

2 ARC Framework Using a Specific M_k-Norm

2.1 ARC Framework

We consider a problem of unconstrained minimization of the form

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

where the objective function $f : \mathbb{R}^n \to \mathbb{R}$ is assumed to be continuously differentiable. The ARC framework [5] can be described as follows: at a given iterate x_k, we define $m_k^Q : \mathbb{R}^n \to \mathbb{R}$ as an approximate second-order Taylor approximation of the objective function f around x_k, i.e.,

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

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 (uniformly bounded from above) of the Hessian of f at x_k. The trial step s_k

approximates the global minimizer of the cubic model $m_k(s) = m_k^Q(s) + \frac{1}{3}\sigma_k\|s\|_{M_k}^3$, i.e.,

$$s_k \approx \arg\min_{s\in\mathbb{R}^n} m_k(s), \tag{3}$$

where $\|.\|_{M_k}$ denotes an iteration dependent scaled norm of the form $\|x\|_{M_k} = \sqrt{x^\top M_k x}$ for all $x \in \mathbb{R}^n$ and M_k is a given SPD matrix. $\sigma_k > 0$ is a dynamic positive parameter that might be regarded as the reciprocal of the TR radius in TR algorithms (see [5]). The parameter σ_k is taking into account the agreement between the objective function f and the model m_k.

To decide whether the trial step is acceptable or not a ratio between the actual reduction and the predicted reduction is computed, as follows:

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

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 $x_{k+1} = x_k + s_k$; otherwise the current iterate is kept unchanged $x_{k+1} = x_k$. We note that, unlike the original ARC [5, 4] where the cubic model is used to evaluate the denominator in (4), in the nowadays works related to ARC, only the quadratic approximation $m_k^Q(s_k)$ is used in the comparison with the actual value of f without the regularization parameter (see [2] for instance). Algorithm 1 gives a detailed description of ARC.

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 $\sigma_{\min} \in]0, \sigma_0]$, set also the constants $0 < \nu_1 \leq 1 < \nu_2$.

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

Compute the step s_k as an approximate solution of (3) such that

$$m_k(s_k) \;\leq\; m_k(s_k^c) \tag{5}$$

where $s_k^c = -\delta_k^c g_k$ and $\delta_k^c = \arg\min_{t>0} m_k(-tg_k)$;

if $\rho_k \geq \eta$ **then**

Set $x_{k+1} = x_k + s_k$ and $\sigma_{k+1} = \max\{\nu_1\sigma_k, \sigma_{\min}\}$;

else

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

end

end

The Cauchy step s_k^c, defined in Algorithm 1, is computationally inexpensive compared to the computational cost of the global minimizer of m_k. The condition (5) on s_k is sufficient for ensuring global convergence of ARC to first-order critical points.

From now on, we will assume that first-order stationarity is not reached yet, meaning that the gradient of the objective function is non null at the current iteration k (i.e., $g_k \neq 0$). Also, $\|\cdot\|$ will denote the vector or matrix ℓ_2-norm, $\mathrm{sgn}(\alpha)$ the sign of a real α, and I_n the identity matrix of size n.

2.2 An LS Algorithm Inspired by the ARC Framework

Using a specific M_k-norm in the definition of the cubic model m_k, we will show that ARC framework (Algorithm 1) behaves as an LS algorithm along the quasi-Newton direction. In a previous work [1], when the matrix B_k is assumed to be positive definite, we showed that the minimizer s_k of the cubic model defined in (3) is getting collinear with the quasi-Newton direction when the matrix M_k is set to be equal to B_k. In this section we generalize our proposed approach to cover the case where the linear system $B_k s = -g_k$ admits an approximate solution and B_k is not necessarily SPD.

Let s_k^Q be an approximate solution of the linear system $B_k s = -g_k$ and assume that such step s_k^Q is not orthogonal with the gradient of the objective function at x_k, i.e., there exists an $\epsilon_d > 0$ such that $|g_k^\top s_k^Q| \geq \epsilon_d \|g_k\| \|s_k^Q\|$. Suppose that there exists an SPD matrix M_k such that $M_k s_k^Q = \frac{\beta_k \|s_k^Q\|^2}{g_k^\top s_k^Q} g_k$ where $\beta_k \in]\beta_{\min}, \beta_{\max}[$ and $\beta_{\max} > \beta_{\min} > 0$, in Theorem 4.1 we will show that such matrix M_k exists. By using the associated M_k-norm in the definition of the cubic model m_k, one can show (see Theorem 3.3) that an approximate stationary point of the subproblem (3) is of the form

$$s_k \;=\; \delta_k s_k^Q, \quad \text{where} \quad \delta_k = \frac{2}{1 - \operatorname{sgn}(g_k^\top s_k^Q)\sqrt{1 + 4\frac{\sigma_k \|s_k^Q\|_{M_k}^3}{|g_k^\top s_k^Q|}}}. \tag{6}$$

For *unsuccessful* iterations in Algorithm 1, since the step direction s_k^Q does not change, the approximate solution of the subproblem, given by (6), can be obtained only by updating the step-size δ_k. This means that the subproblem computational cost of *unsuccessful* iterations is getting straightforward compared to solving the subproblem as required by ARC when the Euclidean norm is used (see e.g., [5]). As a consequence, the use of the proposed M_k-norm in Algorithm 1 will lead to a new formulation of ARC algorithm where the dominant computational cost, regardless the objective function evaluation cost, is the cost of solving a linear system for *successful* iterations. In other words, with the proposed M_k-norm, the ARC algorithm behaves as an LS method with a specific backtracking strategy and an acceptance criteria in the sprite of ARC algorithm, i.e., the step is of the form $s_k = \delta_k s_k^Q$ where the step length $\delta_k > 0$ is chosen such as

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{f(x_k) - m_k^Q(s_k)} \geq \eta \quad \text{and} \quad m_k(s_k) \leq m_k(-\delta_k^{\mathrm{c}} g_k). \tag{7}$$

The step lengths δ_k and δ_k^{c} are computed respectively as follows:

$$\delta_k \;=\; \frac{2}{1 - \operatorname{sgn}(g_k^\top s_k^Q)\sqrt{1 + 4\frac{\sigma_k \beta_k^{3/2} \|s_k^Q\|^3}{|g_k^\top s_k^Q|}}}, \tag{8}$$

and

$$\delta_k^{\mathrm{c}} \;=\; \frac{2}{\frac{g_k^\top B_k g_k}{\|g_k\|^2} + \sqrt{\left(\frac{g_k^\top B_k g_k}{\|g_k\|^2}\right)^2 + 4\sigma_k \chi_k^{3/2} \|g_k\|}} \tag{9}$$

where $\chi_k = \beta_k \left(\frac{5}{2} - \frac{3}{2}\cos(\varpi_k)^2 + 2\left(\frac{1-\cos(\varpi_k)^2}{\cos(\varpi_k)}\right)^2 \right)$ and $\cos(\varpi_k) = \frac{g_k^\top s_k^Q}{\|g_k\|\|s_k^Q\|}$. The M_k-norms of the vectors s_k^Q and g_k in the computation of δ_k and δ_k^c have been substituted using the expressions given in Theorem 4.1. The value of σ_k is set equal to the current value of the regularization parameter as in the original ARC algorithm. For large values of δ_k the decrease condition (7) may not be satisfied. In this case, the value of σ_k is enlarged using an expansion factor $\nu_2 > 1$. Iteratively, the value of δ_k is updated and the acceptance condition (7) is checked again, until its satisfaction.

We referred the ARC algorithm, when the proposed scaled M_k-norm is used, by LS-ARC as it behaves as an LS type method in this case. Algorithm 2 details the final algorithm. We recall again that this algorithm is nothing but ARC algorithm using a specific M_k-norm.

Algorithm 2: LS-ARC algorithm.

Data: select an initial point x_0 and the constants $0 < \eta < 1$, $0 < \epsilon_d < 1$, $0 < \nu_1 \leq 1 < \nu_2$ and $0 < \beta_{\min} < \beta_{\max}$. Set $\sigma_0 > 0$ and $\sigma_{\min} \in]0, \sigma_0]$.
for $k = 1, 2, \ldots$ **do**
  Choose a parameter $\beta_k \in]\beta_{\min}, \beta_{\max}[$;
  Let s_k^Q be an approximate solution of $B_k s = -g_k$ such as $|g_k^\top s_k^Q| \geq \epsilon_d \|g_k\| \|s_k^Q\|$;
  Set δ_k and δ_k^c respectively using (8) and (9);
  while condition (7) is not satisfied **do**
    Set $\sigma_k \leftarrow \nu_2 \sigma_k$ and update δ_k and δ_k^c respectively using (8) and (9);
  end
  Set $s_k = \delta_k s_k^Q$, $x_{k+1} = x_k + s_k$ and $\sigma_{k+1} = \max\{\nu_1 \sigma_k, \sigma_{\min}\}$;
end

Note that in Algorithm 2 the step s_k^Q may not exist or be approximately orthogonal with the gradient g_k. A possible way to overcome this issue can be ensured by modifying the matrix B_k using regularization techniques. In fact, as far as the Hessian approximation is still uniformly bounded from above, the global convergence will still hold as well as a complexity bound of order ϵ^{-2} to drive the norm of the gradient below $\epsilon \in]0, 1[$ (see [5] for instance).

The complexity bound can be improved to be of the order of $\epsilon^{-3/2}$ if a second order version of the algorithm LS-ARC is used, by making B_k equals to the exact Hessian or at least being a good approximation of the exact Hessian (as in Assumption 4.2 of Section 4). In this case, modify the matrix B_k using regularization techniques such that the step s_k^Q approximates the linear system $B_k s = -g_k$ and $|g_k^\top s_k^Q| \geq \epsilon_d \|g_k\| \|s_k^Q\|$ is not trivial anymore. This second order version of the algorithm LS-ARC will be discussed in details in Section 4 where convergence and complexity analysis, when the proposed M_k-norm is used, will be outlined.

3 On the Cubic Model Minimization

In this section, we assume that the linear system $B_k s = -g_k$ has a solution. We will mostly focus on the solution of the subproblem (3) for a given outer iteration k. In particular, we will explicit the condition to impose on the matrix M_k in order to get the solution of the ARC subproblem collinear with the step s_k^Q. Hence, in such case, one can get the solution of the ARC subproblem at a modest computational cost.

The step s_k^Q can be obtained exactly using a direct method if the matrix B_k is not too large. Typically, one can use the LDL^T factorization to solve this linear system. For large scale optimization problems, computing s_k^Q can be prohibitively computationally expensive. We will show that it will be possible to relax this requirement by letting the step s_k^Q be only an approximation of the exact solution using subspace methods.

In fact, when an approximate solution is used and as far as the global convergence of Algorithm 1 is concerned, all what is needed is that the solution of the subproblem (3) yields a decrease in the cubic model which is as good as the Cauchy decrease (as emphasized in condition (5)). In practice, a version of Algorithm 2 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_k influences also the minimization direction. The main idea consists of achieving a further decrease on the cubic model, better than 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 computationally cheap and yields the global convergence of Algorithm 2.

A classical way to approximate the exact solution s_k^Q is by using subspace methods, typically a Krylov subspace method. For that, let $\mathcal{L}_k$ be a subspace of $\mathbb{R}^n$ and l its dimension. Let Q_k denotes an $n \times l$ matrix whose columns form a basis of $\mathcal{L}_k$. Thus for all $s \in \mathcal{L}_k$, we have $s_k = Q_k z_k$, for some $z_k \in \mathbb{R}^l$. In this case, s_k^Q denotes the exact stationary point of the model function m^Q over the subspace $\mathcal{L}_k$ when it exists.

For both cases, exact and inexact, we will assume that the step s_k^Q is not orthogonal with the gradient g_k. In what comes next, we state our assumption on s_k^Q formally as follows:

Assumption 3.1 *The model m_k^Q admits a stationary point s_k^Q such as $|g_k^\top s_k^Q| \geq \epsilon_d \|g_k\| \|s_k^Q\|$ where $\epsilon_d > 0$ is a pre-defined positive constant.*

We define also a Newton-like step s_k^{N} associated with the minimization of the cubic model m_k over the subspace $\mathcal{L}_k$ on the following way, when s_k^Q corresponds to the exact solution of $B_k s = -g_k$ by

$$s_k^{\mathrm{N}} = \delta_k^{\mathrm{N}} s_k^Q, \quad \text{where} \quad \delta_k^{\mathrm{N}} = \arg\min_{\delta \in \mathcal{I}_k} m_k(\delta s_k^Q), \tag{10}$$

where $\mathcal{I}_k = \mathbb{R}_+$ if $g_k^\top s_k^Q < 0$ and $\mathcal{I}_k = \mathbb{R}_-$ otherwise. If s_k^Q is computed using an iterative subspace method, then $s_k^{\mathrm{N}} = Q_k z_k^{\mathrm{N}}$, where z_k^{N} is the Newton-like step , as in (10), associated to the following reduced subproblem:

$$\min_{z \in \mathbb{R}^l} f(x_k) + z^\top Q_k^\top g_k + \frac{1}{2} z^\top Q_k^\top B_k Q_k z + \frac{1}{3} \sigma_k \|z\|^3_{Q_k^\top M_k Q_k}. \tag{11}$$

Theorem 3.1 *Let Assumption 3.1 hold. The Newton-like step s_k^{N} is of the form*

$$s_k^{\mathrm{N}} = \delta_k^{\mathrm{N}} s_k^Q, \quad \textit{where} \quad \delta_k^{\mathrm{N}} = \frac{2}{1 - \mathrm{sgn}(g_k^\top s_k^Q)\sqrt{1 + 4\frac{\sigma_k \|s_k^Q\|^3_{M_k}}{|g_k^\top s_k^Q|}}}. \tag{12}$$

Proof. Consider first the case where the step s_k^Q is computed exactly (i.e., $B_k s_k^Q = -g_k$). In this case, for all $\delta \in \mathcal{I}_k$, one has

$$\begin{aligned} m_k(\delta s_k^Q) - m_k(0) &= \delta g_k^\top s_k^Q + \frac{\delta^2}{2}[s_k^Q]^\top B_k[s_k^Q] + \frac{\sigma|\delta|^3}{3}\|s_k^Q\|_{M_k}^3 \\ &= (g_k^\top s_k^Q)\delta - (g_k^\top s_k^Q)\frac{\delta^2}{2} + (\sigma_k\|s_k^Q\|_{M_k}^3)\frac{|\delta|^3}{3}. \end{aligned} \tag{13}$$

If $g_k^\top s_k^Q < 0$ (hence, $\mathcal{I}_k = \mathbb{R}_+$), we compute the value of the parameter δ_k^N at which the unique minimizer of the above function is attained. Taking the derivative of (13) with respect to δ and equating the result to zero, one gets

$$0 = g_k^\top s_k^Q - (g_k^\top s_k^Q)\delta_k^{\mathrm{N}} + \sigma_k\|s_k^Q\|_{M_k}^3\left(\delta_k^{\mathrm{N}}\right)^2, \tag{14}$$

and thus, since $\delta_k^{\mathrm{N}} > 0$,

$$\delta_k^{\mathrm{N}} = \frac{g_k^\top s_k^Q + \sqrt{\left(g_k^\top s_k^Q\right)^2 - 4\sigma_k(g_k^\top s_k^Q)\|s_k^Q\|_{M_k}^3}}{2\sigma_k\|s_k^Q\|_{M_k}^3} = \frac{2}{1 + \sqrt{1 - 4\frac{\sigma_k\|s_k^Q\|_{M_k}^3}{g_k^\top s_k^Q}}}.$$

If $g_k^\top s_k^Q > 0$ (hence, $\mathcal{I}_k = \mathbb{R}_-$),and again by taking the derivative of (13) with respect to δ and equating the result to zero, one gets

$$0 = g_k^\top s_k^Q - (g_k^\top s_k^Q)\delta_k^{\mathrm{N}} - \sigma_k\|s_k^Q\|_{M_k}^3\left(\delta_k^{\mathrm{N}}\right)^2, \tag{15}$$

and thus, since $\delta_k^{\mathrm{N}} < 0$ in this case,

$$\delta_k^{\mathrm{N}} = \frac{g_k^\top s_k^Q + \sqrt{\left(g_k^\top s_k^Q\right)^2 + 4\sigma_k(g_k^\top s_k^Q)\|s_k^Q\|_{M_k}^3}}{2\sigma_k\|s_k^Q\|_{M_k}^3} = \frac{2}{1 - \sqrt{1 + 4\frac{\sigma_k\|s_k^Q\|_{M_k}^3}{g_k^\top s_k^Q}}}.$$

From both cases, one deduces that $\delta_k^{\mathrm{N}} = \frac{2}{1-\mathrm{sgn}(g_k^\top s_k^Q)\sqrt{1+4\frac{\sigma_k\|s_k^Q\|_{M_k}^3}{|g_k^\top s_k^Q|}}}$.

Consider now the case where s_k^Q is computed using an iterative subspace method. In this cas, one has $s_k^{\mathrm{N}} = Q_k z_k^{\mathrm{N}}$, where z_k^{N} is the Newton-like step associated to the reduced subproblem (11). Hence by applying the first part of the proof (the exact case) to the reduced subproblem (11), it follows that

$$z_k^{\mathrm{N}} = \bar{\delta}_k^{\mathrm{N}} z_k^Q \quad \text{where} \quad \bar{\delta}_k^{\mathrm{N}} = \frac{2}{1 - \mathrm{sgn}((Q_k^\top g_k)^\top z_k^Q)\sqrt{1 + 4\frac{\sigma_k\|z_k^Q\|_{Q_k^\top M_k Q_k}^3}{|(Q_k^\top g_k)^\top z_k^Q|}}},$$

where z_k^Q is a stationary point of the quadratic part of the minimized model in (11). Thus, by

substituting z_k^{N} in the formula $s_k^{\mathrm{N}} = Q_k z_k^{\mathrm{N}}$, one gets

$$\begin{aligned} s_k^{\mathrm{N}} &= Q_k \left(\frac{2}{1 - \mathrm{sgn}((Q_k^\top g_k)^\top z_k^Q)\sqrt{1 + 4\frac{\sigma_k \|z_k^Q\|^3_{Q_k^\top M_k Q_k}}{|(Q_k^\top g_k)^\top z_k^Q|}}} z_k^Q \right) \\ &= \frac{2}{1 - \mathrm{sgn}(g_k^\top Q_k z_k^Q)\sqrt{1 + 4\frac{\sigma_k \|Q_k z_k^Q\|^3_{M_k}}{|g_k^\top Q_k z_k^Q|}}} Q_k z_k^Q \\ &= \frac{2}{1 - \mathrm{sgn}(g_k^\top s_k^Q)\sqrt{1 + 4\frac{\sigma_k \|s_k^Q\|^3_{M_k}}{|g_k^\top s_k^Q|}}} s_k^Q. \end{aligned}$$

■

In general, for ARC algorithm, the matrix M_k can be any arbitrary SPD matrix. Our goal, in this section, is to determine how one can choose the matrix M_k so that the Newton-like step s_k^{N} becomes a stationary point of the subproblem (3). The following theorem gives explicitly the necessary and sufficient condition on the matrix M_k to reach this aim.

Theorem 3.2 *Let Assumption 3.1 hold. The step s_k^{N} is a stationary point for the subproblem (3) if and only if there exists $\theta_k > 0$ such that $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g_k$. Note that $\theta_k = \|s_k^Q\|^2_{M_k}$.*

Proof. Indeed, in the exact case, if we suppose that the step s_k^{N} is a stationary point of the subproblem (3), this means that

$$\nabla m_k(s_k^{\mathrm{N}}) = g_k + B_k s_k^{\mathrm{N}} + \sigma_k \|s_k^{\mathrm{N}}\|_{M_k} M_k s_k^{\mathrm{N}} = 0, \tag{16}$$

In another hand, $s_k^{\mathrm{N}} = \delta_k^{\mathrm{N}} s_k^Q$ where δ_k^{N} is solution of $g_k^\top s_k^Q - (g_k^\top s_k^Q)\delta_k^{\mathrm{N}} + \sigma_k \|s_k^Q\|^3_{M_k} |\delta_k^{\mathrm{N}}| \delta_k^{\mathrm{N}} = 0$ (such equation can be deduced from (14) and (15)). Hence, we obtain that

$$\begin{aligned} 0 &= \nabla m_k(s_k^{\mathrm{N}}) = g_k - \delta_k^{\mathrm{N}} g_k + \sigma_k |\delta_k^{\mathrm{N}}| \delta_k^{\mathrm{N}} \|s_k^Q\|_{M_k} M_k s_k^Q \\ &= \left(1 - \delta_k^{\mathrm{N}}\right) g_k + \left(\frac{\sigma_k \|s_k^Q\|^3_{M_k}}{g_k^\top s_k^Q} |\delta_k^{\mathrm{N}}| \delta_k^{\mathrm{N}} \right) \left(\frac{g_k^\top s_k^Q}{\|s^Q\|^2_{M_k}} M_k s_k^Q \right) \\ &= \left(\frac{\sigma_k \|s_k^Q\|^3_{M_k}}{g_k^\top s_k^Q} |\delta_k^{\mathrm{N}}| \delta_k^{\mathrm{N}} \right) \left(g_k - \frac{g_k^\top s_k^Q}{\|s_k^Q\|^2_{M_k}} M_k s_k^Q \right). \end{aligned}$$

Equivalently, we conclude that $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g_k$ where $\theta_k = \|s_k^Q\|^2_{M_k} > 0$. A similar proof applies when a subspace method is used to compute s_k^Q. ■

The key condition to ensure that the ARC subproblem stationary point is equal to the Newton-like step s_k^{N}, is the choice of the matrix M_k which satisfies the following secant-like equation $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g_k$ for a given $\theta_k > 0$. The existence of such matrix M_k is not problematic as far as Assumption 3.1 holds. In fact, Theorem 4.1 will explicit a range of $\theta_k > 0$ for which the matrix M_k exists. Note that in the formula of s_k^{N}, such matrix is used only through the

computation of the M_k-norm of s_k^Q. Therefore an explicit formula of the matrix M_k is not needed, and only the value of $\theta_k = \|s_k^Q\|_{M_k}^2$ suffices for the computations.

When the matrix M_k satisfies the desired properties (as in Theorem 3.2), one is ensured that s_k^{N} is a stationary point for the model m_k. However, ARC algorithm imposes on the approximate step to satisfy the Cauchy decrease given by (5), and such condition is not guaranteed by s_k^{N} as the model m_k may be non-convex. In the next theorem, we show that for a sufficiently large σ_k, s_k^{N} is getting the global minimizer of m_k and thus satisfying the Cauchy decrease is not an issue anymore.

Theorem 3.3 *Let Assumption 3.1 hold. Let M_k be an SPD matrix which satisfies $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g_k$ for a fixed $\theta_k > 0$. If the matrix $Q_k^\top (B_k + \sigma_k \|s_k^{\mathrm{N}}\|_{M_k} M_k) Q_k$ is positive definite, then the step s_k^{N} is the unique minimizer of the subproblem (3) over the subspace $\mathcal{L}_k$.*

Proof. Indeed, when s^Q is computed exactly (i.e., $Q_k = I_n$ and $\mathcal{L}_k = \mathbb{R}^n$), then using [8, Theorem 3.1] one has that, for a given vector s_k^*, it is a global minimizer of m_k if and only if it satisfies

$$(B_k + \lambda_k^* M_k) s_k^* = -g_k$$

where $B_k + \lambda_k^* M_k$ is positive semi-definite matrix and $\lambda_k^* = \sigma_k \|s_k^*\|_{M_k}$. Moreover, if $B_k + \lambda_k^* M_k$ is positive definite, s_k^* is unique.

Since $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g_k$, by applying Theorem 3.2, we see that

$$(B_k + \lambda_k^{\mathrm{N}} M_k) s_k^{\mathrm{N}} = -g_k$$

with $\lambda_k^{\mathrm{N}} = \sigma_k \|s_k^{\mathrm{N}}\|_{M_k}$. Thus, if we assume that $B_k + \lambda_k^{\mathrm{N}} M_k$ is positive definite matrix, then s_k^{N} is the unique global minimizer of the subproblem (3).

Consider now, the case where s_k^Q is computed using a subspace method, since $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g_k$ one has that $Q_k^\top M_k Q_k z_k^Q = \frac{\theta_k}{(Q_k^\top g_k)^\top z_k^Q} Q_k^\top g_k$. Hence, if we suppose that the matrix $Q_k^\top (B_k + \lambda_k^{\mathrm{N}} M_k) Q_k$ is positive definite, by applying the same proof of the exact case to the reduced subproblem (11), we see that the step z_k^{N} is the unique global minimizer of the subproblem (11). We conclude that $s_k^{\mathrm{N}} = Q_k z_k^{\mathrm{N}}$ is the global minimizer of the subproblem (3) over the subspace $\mathcal{L}_k$. ■

Theorem 3.3 states that the step s_k^{N} is the global minimizer of the cubic model m_k over the subspace $\mathcal{L}_k$ as far as the matrix $Q_k^\top (B_k + \sigma_k \|s_k^{\mathrm{N}}\|_{M_k} M_k) Q_k$ is positive definite, where $\lambda_k^{\mathrm{N}} = \sigma_k \|s_k^{\mathrm{N}}\|_{M_k}$. Note that

$$\lambda_k^{\mathrm{N}} \;=\; \sigma_k \|s_k^{\mathrm{N}}\|_{M_k} = \frac{2\sigma_k \|s_k^Q\|_{M_k}}{\left| 1 - \mathrm{sgn}(g_k^\top s_k^Q) \sqrt{1 + 4 \frac{\sigma_k \|s_k^Q\|_{M_k}^3}{|g_k^\top s_k^Q|}} \right|} \to +\infty \quad \text{as } \sigma_k \to \infty.$$

Thus, since M_k is an SPD matrix and the regularization parameter σ_k is increased for unsuccessful iterations in Algorithm 1, the positive definiteness of matrix $Q_k^\top (B_k + \sigma_k \|s_k^{\mathrm{N}}\|_{M_k} M_k) Q_k$ is guaranteed after finitely many unsuccessful iterations. In other words, one would have insurance that s_k^{N} will satisfy the Cauchy decrease after a certain number of unsuccessful iterations.

4 Complexity Analysis of the LS-ARC Algorithm

For the well definiteness of the algorithm LS-ARC, one needs first to show that the proposed M_k-norm is uniformly equivalent to the Euclidean norm. The next theorem gives a range of choices for the parameter θ_k to ensure the existence of an SPD matrix M_k such as $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g_k$ and the M_k-norm is uniformly equivalent to the ℓ_2-norm.

Theorem 4.1 *Let Assumption 3.1 hold. If*

$$\theta_k = \beta_k \|s_k^Q\|^2 \quad \textit{where } \beta_k \in]\beta_{\min}, \beta_{\max}[\quad \textit{and} \quad \beta_{\max} > \beta_{\min} > 0, \tag{17}$$

then there exists an SPD matrix M_k such as

i) $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g_k$,

ii) the M_k-norm is uniformly equivalent to the ℓ_2-norm on $\mathbb{R}^n$ and for all $x \in \mathbb{R}^n$, one has

$$\frac{\sqrt{\beta_{\min}}}{\sqrt{2}} \|x\| \leq \|x\|_{M_k} \leq \frac{\sqrt{2\beta_{\max}}}{\epsilon_d} \|x\|. \tag{18}$$

iii) Moreover, one has $\|s_k^Q\|_{M_k}^2 = \beta_k \|s_k^Q\|^2$ and $\|g_k\|_{M_k}^2 = \chi_k \|g_k\|^2$ where $\chi_k = \beta_k \left(\frac{5}{2} - \frac{3}{2}\cos(\varpi_k)^2 + 2 \left(\frac{1-\cos(\varpi$
and $\cos(\varpi_k) = \frac{g_k^\top s_k^Q}{\|g_k\| \|s_k^Q\|}$

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

$$\frac{g_k}{\|g_k\|} = \cos(\varpi_k) \bar{s}_k^Q + \sin(\varpi_k) \bar{g}_k. \tag{19}$$

For a given $\theta_k = \beta_k \|s_k^Q\|^2$ where $\beta_k \in]\beta_{\min}, \beta_{\max}[$ and $\beta_{\max} > \beta_{\min} > 0$, one would like to construct an SPD matrix M_k such as $M_k s_k^Q = \frac{\theta_k g_k}{g_k^\top s_k^Q}$, hence

$$\begin{aligned} M_k \bar{s}_k^Q = \frac{\theta_k g_k}{g_k^\top s_k^Q \|s_k^Q\|} &= \frac{\theta_k \|g_k\|}{g_k^\top s_k^Q \|s_k^Q\|} \left(\cos(\varpi_k) \bar{s}_k^Q + \sin(\varpi_k) \bar{g}_k \right) \\ &= \beta_k \bar{s}_k^Q + \beta_k \tan(\varpi_k) \bar{g}_k. \end{aligned}$$

Using the symmetric structure of the matrix M_k, let γ_k be a positive parameter such as

$$M_k = \left[\bar{s}_k^Q, \bar{g}_k \right] N_k \left[\bar{s}_k^Q, \bar{g}_k \right]^\top \text{ where } N_k = \begin{bmatrix} \beta_k & \beta_k \tan(\varpi_k) \\ \beta_k \tan(\varpi_k) & \gamma_k \end{bmatrix}.$$

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

$$\lambda^2 - (\beta_k + \gamma_k) \lambda + \beta_k \gamma_k - (\beta_k \tan(\varpi_k))^2 = 0,$$

hence

$$\lambda_k^{\min} = \frac{(\beta_k + \gamma_k) - \sqrt{\vartheta_k}}{2} \quad \text{and} \quad \lambda_k^{\max} = \frac{(\beta_k + \gamma_k) + \sqrt{\vartheta_k}}{2},$$

where $\vartheta_k = (\beta_k - \gamma_k)^2 + 4\left(\beta_k \tan(\varpi_k)\right)^2$. Note that both eigenvalues are monotonically increasing as functions of γ_k.

One may choose $\lambda_k^{\min}$ to be equal to $\frac{1}{2}\beta_k = \frac{1}{2}\frac{\theta_k}{\|s_k^Q\|^2}$, therefore $\lambda_k^{\min} > \frac{1}{2}\beta_{\min}$ is uniformly bounded away from zero. In this case, from the expression of $\lambda_k^{\min}$, we deduce that $\gamma_k = 2\beta_k \tan(\varpi_k)^2 + \beta_k/2$ and

$$\begin{aligned}
\lambda_k^{\max} &= \frac{3}{4}\beta_k + \beta_k \tan(\varpi_k)^2 + \sqrt{\frac{1}{16}\beta_k^2 + \frac{1}{2}\beta_k^2 \tan(\varpi_k)^2 + \beta_k^2 \tan(\varpi_k)^4} \\
&= \beta_k \left(\frac{3}{4} + \tan(\varpi_k)^2 + \sqrt{\frac{1}{16} + \frac{1}{2}\tan(\varpi_k)^2 + \tan(\varpi_k)^4} \right) \\
&= \beta_k \left(1 + 2\tan(\varpi_k)^2\right).
\end{aligned}$$

From Assumption 3.1, i.e., $|g_k^\top s_k^Q| \geq \epsilon_d \|g_k\| \|s_k^Q\|$ where $\epsilon_d > 0$, one has $\tan(\varpi_k)^2 \leq \frac{1-\epsilon_d^2}{\epsilon_d^2}$. Hence,

$$\lambda_k^{\max} \leq \beta_{\max}\left(1 + 2\frac{1-\epsilon_d^2}{\epsilon_d^2}\right) \leq \frac{2\beta_{\max}}{\epsilon_d^2}$$

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

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

where $D = \operatorname{diag}(d_3, \ldots, d_n) \in \mathbb{R}^{(n-2)\times(n-2)}$ with positive diagonal entrees independent from k . One concludes that for all $\theta_k = \beta_k \|s_k^Q\|^2$ where $\beta_k \in]\beta_{\min}, \beta_{\max}[$ and $\beta_{\max} > \beta_{\min} > 0$, the eigenvalue of the constructed M_k are uniformly bounded away from zero and from above, hence the scaled M_k-norm is uniformly equivalent to the ℓ_2-norm on $\mathbb{R}^n$ and for all $x \in \mathbb{R}^n$, one has

$$\frac{\sqrt{\beta_{\min}}}{\sqrt{2}}\|x\| \leq \sqrt{\lambda_k^{\min}}\|x\| \leq \|x\|_{M_k} \leq \sqrt{\lambda_k^{\max}}\|x\| \leq \frac{\sqrt{2\beta_{\max}}}{\epsilon_d}\|x\|.$$

By multiplying $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g_k$ from both sides by s_k^Q, one gets

$$\|s_k^Q\|_{M_k}^2 = \theta_k = \beta_k \|s_k^Q\|^2.$$

Moreover, using (19) and (20), one has

$$\begin{aligned}
\|g_k\|_{M_k}^2 &= \|g_k\|^2 \left(\cos(\varpi_k)\bar{s}_k^Q + \sin(\varpi_k)\bar{g}_k\right)^\top \left(\cos(\varpi_k) M_k \bar{s}_k^Q + \sin(\varpi_k) M_k \bar{g}_k\right) \\
&= \|g_k\|^2 \left(\frac{\theta_k \cos(\varpi_k)^2}{\|s_k^Q\|^2} + \gamma_k \sin(\varpi_k)^2 + 2\sin(\varpi_k)\cos(\varpi_k)\frac{\theta_k \tan(\varpi_k)^2}{\|s_k^Q\|^2} \right) \\
&= \beta_k \|g_k\|^2 \left(\cos(\varpi_k)^2 + \frac{5}{2}\sin(\varpi_k)^2 + 2\sin(\varpi_k)^2 \tan(\varpi_k)^2 \right) \\
&= \beta_k \|g_k\|^2 \left(\frac{5}{2} - \frac{3}{2}\cos(\varpi_k)^2 + 2\left(\frac{1-\cos(\varpi_k)^2}{\cos(\varpi_k)}\right)^2 \right).
\end{aligned}$$

■

A direct consequence of Theorem 4.1 is that, by choosing $\theta_k > 0$ of the form $\beta_k \|s_k^Q\|^2$ where $\beta_k \in]\beta_{\min}, \beta_{\max}[$ and $\beta_{\max} > \beta_{\min} > 0$ during the application of LS-ARC algorithm, the global convergence and complexity bounds of LS-ARC algorithm can be derived straightforwardly from the classical ARC analysis [4]. In fact, as far as the objective function f is continuously differentiable, its gradient is Lipschitz continuous, and its approximated Hessian B_k is bounded for all iterations (see [4, Assumptions AF1, AF4, and AM1]), the LS-ARC algorithm is globally convergent and will required at most a number of iterations of order ϵ^{-2} to produce a point x_ϵ with $\|\nabla f(x_\epsilon)\| \leq \epsilon$ [4, Corollary 3.4].

In what comes next, we assume the following on the objective function f:

Assumption 4.1 *Assume that f is twice continuously differentiable with Lipschitz continuous Hessian, i.e., there exists a constant $L \geq 0$ such that for all $x, y \in \mathbb{R}^n$ one has*

$$\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq L\|x - y\|.$$

When the matrix B_k is set to be equal to the exact Hessian of the problem and under Assumption 4.1, one can improve the function-evaluation complexity to be $\mathcal{O}(\epsilon^{-3/2})$ for ARC algorithm by imposing, in addition to the Cauchy decrease, another termination condition during the computation of the trial step s_k (see [4, 2]). Such condition is of the form

$$\|\nabla m_k(s_k)\| \quad \leq \quad \zeta\|s_k\|^2, \tag{20}$$

where $\zeta > 0$ is a given constant chosen at the start of the algorithm.

When only an approximation of the Hessian is available during the application Algorithm 1, an additional condition has to be imposed on the Hessian approximation B_k in order to ensure an optimal complexity of order $\epsilon^{-3/2}$. Such condition is often considered as (see [4, Assumption AM.4]):

Assumption 4.2 *The matrix B_k approximate the Hessian $\nabla^2 f(x_k)$ in the sense that*

$$\|(\nabla^2 f(x_k) - B_k)s_k\| \leq C\|s_k\|^2 \tag{21}$$

for all $k \geq 0$ and for some constant $C > 0$.

Similarly, for LS-ARC algorithm, the complexity bound can be improved to be of the order of $\epsilon^{-3/2}$ if one includes the two following requirements (a) the s_k satisfies the criterion condition (20) and (b) the Hessian approximation matrix B_k has to satisfy Assumption 4.2. When our proposed M_k-norm is used, the termination condition (20) imposed on the cubic model m_k can be expressed only in terms of s_k^Q and ∇m_k^Q. The latter condition will be required in the LS-ARC algorithm at each iteration to ensure that it takes at most $\mathcal{O}(\epsilon^{-3/2})$ iterations to reduce the gradient norm below ϵ. Such result is given in the following proposition

Proposition 4.1 *Let Assumption 3.1 hold. Let $\theta_k = \beta_k \|s_k^Q\|^2$ where $\beta_k \in]\beta_{\min}, \beta_{\max}[$ and $\beta_{\max} > \beta_{\min} > 0$. Then imposing the condition (20) in Algorithm 2 is equivalent to the following condition*

$$\|\nabla m_k^Q(s_k^Q)\| \quad \leq \quad \frac{2\,\mathrm{sgn}(g_k^\top s_k^Q)\zeta}{-1 + \mathrm{sgn}(g_k^\top s_k^Q)\sqrt{1 + 4\frac{\sigma_k \theta_k^{3/2}}{|g_k^\top s_k^Q|}}} \|s_k^Q\|^2. \tag{22}$$

Proof. Since Assumption 3.1 holds and $\theta_k = \beta_k \|s_k^Q\|^2$ as in (17), Theorem 4.1 implies the existence of an SPD matrix M_k such that $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g_k$. Using such M_k-norm, an approximate solution of the cubic model m_k is of the form $s_k = \delta_k s_k^Q$ where δ_k is solution of $g_k^\top s_k^Q - (g_k^\top s_k^Q)\delta_k + \sigma_k \|s_k^Q\|_{M_k}^3 |\delta_k| \delta_k = 0$. Hence,

$$\begin{aligned} \nabla m_k(s_k) &= g_k + B_k s_k + \sigma_k \|s_k\|_{M_k} M_k s_k \\ &= g_k + \delta_k B_k s_k^Q + \sigma_k |\delta_k| \delta_k \|s_k^Q\|_{M_k} M_k s_k^Q. \end{aligned}$$

Since $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g_k$ with $\theta_k = \|s_k^Q\|_{M_k}^2$, one has

$$\begin{aligned} \nabla m_k(s_k) &= g_k + \delta_k B_k s_k^Q + \frac{\sigma_k |\delta_k| \delta_k \|s_k^Q\|_{M_k}^3}{g_k^\top s_k^Q} g_k \\ &= \left(1 + \frac{\sigma_k |\delta_k| \delta_k \|s_k^Q\|_{M_k}^3}{g_k^\top s_k^Q}\right) g_k + \delta_k B_k s_k^Q. \end{aligned}$$

From the fact that $g_k^\top s_k^Q - (g_k^\top s_k^Q)\delta_k + \sigma_k \|s_k^Q\|_{M_k}^3 |\delta_k| \delta_k = 0$, one deduces

$$\nabla m_k(s_k) = \delta_k \left(g_k + B_k s_k^Q\right) = \delta_k \nabla m_k^Q(s_k^Q).$$

Hence, the condition (20) is being equivalent to

$$\|\nabla m_k^Q(s_k^Q)\| \leq \frac{\zeta}{\delta_k} \|s_k\|^2 = \zeta |\delta_k| \|s_k^Q\|^2.$$

■

We note that the use of an exact solver to compute s_k^Q implies that the condition (22) will be automatically satisfied for a such iteration. Moreover, 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 (22) 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.

Algorithm 3 summarized a second-order variant of LS-ARC, referred here as LS-ARC$_{(\mathrm{s})}$, which is guaranteed to have an improved iteration worst-case complexity of order $\epsilon^{-3/2}$ (see Theorem 4.2).

The Hessian approximation B_k (as required by Assumption 4.2) involves s_k, hence finding a new matrix B_k so that the new s_k^Q satisfies Assumption 3.1 using regularization techniques is not trivial as s_k is unknown at this stage. A possible way to satisfy Assumption 4.2 without modifying B_k is by choosing s_k^Q as the first iterate to satisfy (22) when using a subspace method to solve the linear system $B_k s = -g_k$. Then, one checks if s_k^Q satisfies Assumption 3.1 or not. If s_k^Q violates the latter assumption, one runs further iterations of the subspace method until Assumption 3.1 will be satisfied. If the subspace method ends and Assumption 3.1 is still violated, one would restore such assumption by minimizing the cubic model using the ℓ_2-norm until a successful outer iteration is found.

For the sake of illustration, consider the minimization of the objective function $f(x, y) = x^2 - y^2$ for all $(x, y) \in \mathbb{R}^2$, starting from $x_0 = (1, 1)$, with $\sigma_0 = 1$, and B_k being the exact Hessian

Algorithm 3: LS-ARC$_{(s)}$ Algorithm.

In each iteration k of Algorithm 2:

Let s_k^Q be an approximate solution of $B_k s = -g_k$ such as $|g_k^\top s_k^Q| \geq \epsilon_d \|g_k\| \|s_k^Q\|$ and the termination condition

$$\|\nabla m_k^Q(s_k^Q)\| \leq \frac{2\,\mathrm{sgn}(g_k^\top s_k^Q)\zeta}{-1+\mathrm{sgn}(g_k^\top s_k^Q)\sqrt{1+4\frac{\sigma_k \beta_k^{3/2}\|s_k^Q\|^3}{|g_k^\top s_k^Q|}}}\|s_k^Q\|^2$$

is satisfied for a given constant $\zeta > 0$ chosen at the beginning of the algorithm.

of the problem during the application of the algorithm. One starts by checking if $s_0^Q = (-1,-1)$ (i.e., the exact solution of the linear system $B_0 s = -g_0$) is a sufficient descent direction or not. Since the slop $g_0^\top s_0^Q$ is equal to zero for this example, the algorithm has to switch to the ℓ_2-norm and thus s_0 will be set as a minimizer of the cubic model with the ℓ_2-norm to define the cubic regularization term. Using a subproblem solver (in our case the `GLRT` solver from `GALAHAD` [12], more details are given in Section 6), one finds the step $s_0 = (-0.4220, 2.7063)$ and the point $x_1 = x_0 + s_0$ is accepted (with $f(x_1) = -13.4027$). Computing the new gradient $g_1 = (1.1559, -7.4126)$ and the quasi-Newton direction $s_1^Q = (-0.5780, -3.7063)$, one has $|g_1^\top s_1^Q| = 20.5485 \geq \epsilon_d \|g_1\| \|s_1^Q\| = 0.0205$ where $\epsilon_d = 10^{-3}$ (hence Assumption 3.1 holds). We perform then our proposed LS strategy along the direction s_1^Q to obtain the step s_1. For this example, except the first one, all the remaining iterations k satisfy the condition $|g_k^\top s_k^Q| \geq \epsilon_d \|g_k\| \|s_k^Q\|$. We note that the regarded minimization problem is unbounded from below, hence f decreases to $-\infty$ during the application of the algorithm.

LS strategies require in general to have a sufficient descent direction for each iteration, it seems then natural that one may need to choose ϵ_d to be large (close to 1) to target good performance. However, during the application of LS-ARC$_{(s)}$ and to satisfy Assumption 3.1 (without modifying the matrix B_k), one may be encouraged to use an ϵ_d small. In what comes next, we will give a detailed complexity analysis of the LS-ARC$_{(s)}$ algorithm in this case. In particular, we will explicit how the complexity bound will depend on the choice of the constant ϵ_d. The following results are obtained from [2, Lemma 2.1] and [2, Lemma 2.2]:

Lemma 4.1 *Let Assumptions 4.1 and 3.1 hold and consider Algorithm 3. Then for all $k \geq 0$, one has*

$$f(x_k) - m_k^Q(s_k) \geq \frac{\sigma_k}{3}\|s_k\|_{M_k}^3, \tag{23}$$

and

$$\sigma_k \leq \sigma_{\max} := \max\left\{\sigma_0, \frac{3\nu_2 L}{2(1-\eta)}\right\}. \tag{24}$$

The next lemma is an adaptation of [2, Lemma 2.3] when the proposed M_k-norm is used in the ARC framework.

Lemma 4.2 *Let Assumptions 4.1, 4.2 and 3.1 hold. Consider Algorithm 3 with* $\theta_k = \beta_k \|s_k^Q\|^2$ *where* $\beta_k \in]\beta_{\min}, \beta_{\max}[$ *and* $\beta_{\max} > \beta_{\min} > 0$. *Then for all* $k \geq 0$

$$\|s_k\| \geq \left(\frac{\|g_{k+1}\|}{L + C + 2\sqrt{2}\sigma_{\max}\beta_{\max}^{3/2}\epsilon_d^{-3} + \zeta} \right)^{\frac{1}{2}}.$$

Proof. Indeed, using Assumptions 4.1 and 4.2 within Taylor expansion, one has

$$\begin{aligned}
\|g_{k+1}\| &\leq \|g_{k+1} - \nabla m_k(s_k)\| + \|\nabla m_k(s_k)\| \\
&\leq \|g_{k+1} - g_k - B_k s_k - \sigma_k \|s_k\|_{M_k} M_k s_k\| + \zeta \|s_k\|^2 \\
&\leq \|g_{k+1} - g_k - \nabla^2 f(x_k) s_k\| + \|(\nabla^2 f(x_k) - B_k) s_k\| + \sigma_k \|M_k\|^{3/2} \|s_k\|^2 + \zeta \|s_k\|^2 \\
&\leq L\|s_k\|^2 + C\|s_k\|^2 + (\sigma_k \|M_k\|^{3/2} + \zeta)\|s_k\|^2.
\end{aligned}$$

Using (24), one has

$$\|g_{k+1}\| \leq (L + C + \sigma_{\max}\|M_k\|^{3/2} + \zeta)\|s_k\|^2.$$

Since Assumption 4.2 holds and $\theta_k = \beta_k \|s_k^Q\|^2$ where $\beta_k \in]\beta_{\min}, \beta_{\max}[$, then using Theorem 4.1, the matrix M_k norm is bounded from above by $2\beta_{\max}\epsilon_d^{-2}$. Hence,

$$\|g_{k+1}\| \leq \left(L + C + 2\sqrt{2}\sigma_{\max}\beta_{\max}^{3/2}\epsilon_d^{-3} + \zeta \right) \|s_k\|^2.$$

■

Theorem 4.2 *Let Assumptions 4.1, 4.2 and 3.1 hold. Consider Algorithm 3 with* $\theta_k = \beta_k \|s_k^Q\|^2$ *where* $\beta_k \in]\beta_{\min}, \beta_{\max}[$ *and* $\beta_{\max} > \beta_{\min} > 0$. *Then, given an* $\epsilon > 0$, *Algorithm 3 needs at most*

$$\left\lfloor \kappa_s(\epsilon_d) \frac{f(x_0) - f_{low}}{\epsilon^{3/2}} \right\rfloor$$

iterations to produce an iterate x_ϵ *such that* $\|\nabla f(x_\epsilon)\| \leq \epsilon$ *where* f_{low} *is a lower bound on* f *and* $\kappa_s(\epsilon_d)$ *is given by*

$$\kappa_s(\epsilon_d) = \frac{6\sqrt{2}\left(L + C + 2\sqrt{2}\sigma_{\max}\beta_{\max}^{3/2}\epsilon_d^{-3} + \zeta \right)^{3/2}}{\eta\sigma_{\min}\beta_{\min}^{3/2}}.$$

Proof. Indeed, at each iteration of Algorithm 3, one has

$$\begin{aligned}
f(x_k) - f(x_k + s_k) &\geq \eta(f(x_k) - m_k^Q(s_k)) \\
&\geq \frac{\eta\sigma_k}{3}\|s_k\|_{M_k}^3 \\
&\geq \frac{\eta\sigma_{\min}\beta_{\min}^{3/2}}{6\sqrt{2}}\|s_k\|^3 \\
&\geq \frac{\eta\sigma_{\min}\beta_{\min}^{3/2}}{6\sqrt{2}\left(L + C + 2\sqrt{2}\sigma_{\max}\beta_{\max}^{3/2}\epsilon_d^{-3} + \zeta \right)^{3/2}}\|g_{k+1}\|^{3/2} \\
&\geq \frac{\eta\sigma_{\min}\beta_{\min}^{3/2}}{6\sqrt{2}\left(L + C + 2\sqrt{2}\sigma_{\max}\beta_{\max}^{3/2}\epsilon_d^{-3} + \zeta \right)^{3/2}}\epsilon^{3/2},
\end{aligned}$$

by using (7), (23), (18), (25), and the fact that $\|g_{k+1}\| \geq \epsilon$ before termination. Thus we deduce for all iterations as long as the stopping criterion does not occur

$$\begin{aligned} f(x_0) - f(x_{k+1}) &= \sum_{j=0}^{k} f(x_j) - f(x_{j+1}) \\ &\geq (k+1) \frac{\eta \sigma_{\min} \sqrt{\beta_{\min}}}{3\sqrt{2}\left(L + C + 2\sqrt{2}\sigma_{\max}\beta_{\max}^{3/2}\epsilon_d^{-3} + \zeta\right)^{3/2}} \epsilon^{3/2}. \end{aligned}$$

Hence, the required number of iterations to produce an iterate x_ϵ such that $\|\nabla f(x_\epsilon)\| \leq \epsilon$ is given as follow

$$k+1 \leq \frac{6\sqrt{2}\left(L + C + 2\sqrt{2}\sigma_{\max}\beta_{\max}^{3/2}\epsilon_d^{-3} + \zeta\right)^{3/2}}{\eta \sigma_{\min} \beta_{\min}^{3/2}} \frac{f(x_0) - f_{\text{low}}}{\epsilon^{3/2}}$$

where f_{low} is a lower bound on f. Thus the proof is completed. ■ We note that $\kappa_{\text{S}}(\epsilon_d)$ can be large for small values of ϵ_d. Hence, although the displayed worst-case complexity bound is of order $\epsilon^{-3/2}$, the latter can be worse than it appears to be if the value of ϵ_d is very small (i.e., the chosen direction is almost orthogonal with the gradient). Such result seems coherent regarding the LS algorithm strategy where it is required to have a sufficient descent direction (i.e., an ϵ_d sufficiently large).

5 TR Algorithm Using a Specific M_k-Norm

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 p_k by approximately solving

$$\min_{p \in \mathbb{R}^n} \quad m_k^Q(p) \qquad \text{s. t.} \quad \|p\|_{M_k} \leq \Delta_k, \tag{25}$$

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

Once the trial step p_k is determined, the objective function is computed at $x_k + p_k$ 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 $x_k + p_k$ 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 $\rho_k \geq \eta$, and *unsuccessful* otherwise. Algorithm 4 gives a detailed description of a basic TR algorithm.

Note that for the TR subproblem, the solution we are looking for lies either interior to the trust region, that is $\|p_k\|_{M_k} < \Delta_k$, or on the boundary, $\|p_k\|_{M_k} = \Delta_k$. If the solution is interior, the solution p_k is the unconstrained minimizer of the quadratic model m_k^Q. Such scenario can only happen if m_k^Q is convex. In the non convex case a solution lies on the boundary of the trust

Algorithm 4: TR algorithm.

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

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

Compute the step p_k as an approximate solution of (25) such that

$$m_k^Q(p_k) \quad \leq \quad m_k(p_k^{\mathrm{c}}) \tag{26}$$

where $p_k^{\mathrm{c}} = -\alpha_k^{\mathrm{c}} g_k$ and $\alpha_k^{\mathrm{c}} = \arg \min_{0<t\leq \frac{\Delta_k}{\|g_k\|_{M_k}}} m_k^Q(-tg_k)$;

if $\rho_k \geq \eta$ **then**

Set $x_{k+1} = x_k + p_k$ and $\Delta_{k+1} = \min\{\tau_2 \Delta_k, \Delta_{\max}\}$;

else

Set $x_{k+1} = x_k$ and $\Delta_{k+1} = \tau_1 \Delta_k$;

end

end

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 m_k^Q. 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.

In this section, we will assume that the approximated solution s_k^Q of the linear system $B_k s = -g_k$ is computed exactly. Using similar arguments as for ARC algorithm, one can extend the obtained results when a truncated step is used in the TR algorithm. Under Assumption 3.1, we will call the Newton-like step associated with the TR subproblem the vector of the following form

$$p_k^{\mathrm{N}} = \alpha_k^{\mathrm{N}} s_k^Q, \quad \text{where } \alpha_k^{\mathrm{N}} = \arg \min_{\alpha \in \mathcal{R}_k} m_k^Q(\alpha s_k^Q), \tag{27}$$

where $\mathcal{R}_k =]0, \frac{\Delta_k}{\|s_k^Q\|_{M_k}}]$ if $g_k^\top s_k^Q < 0$ and $\mathcal{R}_k = [-\frac{\Delta_k}{\|s_k^Q\|_{M_k}}, 0[$ otherwise.

Similarly to ARC algorithm, one has the following results:

Theorem 5.1 *Let Assumption 3.1 hold.*

1. *The Newton-like step (27) is of the following form:*

$$p_k^{\mathrm{N}} \quad = \quad \alpha_k^{\mathrm{N}} s_k^Q, \quad \textit{where} \quad \alpha_k^{\mathrm{N}} = \min\left\{1, -\operatorname{sgn}(g_k^\top s_k^Q) \frac{\Delta}{\|s_k^Q\|_{M_k}}\right\}. \tag{28}$$

2. *When it lies on the border of the trust region p_k^{N} is a stationary point of the subproblem (25) if and only if $M_k s_k^Q = \frac{\theta}{g_k^\top s_k^Q} g_k$ where $\theta_k = \|s_k^Q\|_{M_k}^2$.*

3. *Let $\lambda_k^{\mathrm{N}} = \frac{g_k^\top s_k^Q}{\theta_k}\left(1 + \operatorname{sgn}(g_k^\top s_k^Q)\frac{\|s_k^Q\|_{M_k}}{\Delta_k}\right)$ and assume that p_k^{N} lies on the border of the trust-region. Then, if the matrix $B_k + \lambda_k^{\mathrm{N}} M_k$ is positive definite, the step p_k^{N} will be the unique minimizer of the subproblem (25) over the subspace $\mathcal{L}_k$.*

Proof. 1. To calculate the Newton-like step p_k^{N}, we first note, for all $\alpha \in \mathcal{R}_k$

$$\begin{aligned} m_k^Q(\alpha s_k^Q) - m_k^Q(0) &= \alpha g_k^\top s_k^Q + \frac{\alpha^2}{2}[s_k^Q]^\top B_k[s_k^Q] \\ &= (g_k^\top s_k^Q)\alpha - (g_k^\top s_k^Q)\frac{\alpha^2}{2}. \end{aligned} \tag{29}$$

Consider the case where the curvature model along the Newton direction is positive, that is when $g_k^\top s_k^Q < 0$, (i.e., $\mathcal{R}_k =]0, \frac{\Delta_k}{\|s_k^Q\|_{M_k}}]$) and compute the value of the parameter α at which the unique minimizer of (29) is attained. Let α_k^* denotes this optimal parameter. Taking the derivative of (29) with respect to α and equating the result to zero, one has $\alpha_k^* = 1$. Two sub-cases may then occur. The first is when this minimizer lies within the trust region (i.e., $\alpha_k^* \|s_k^Q\|_{M_k} \leq \Delta_k$), then

$$\alpha_k^{\mathrm{N}} = 1.$$

If $\alpha_k^* \|s_k^Q\|_{M_k} > \Delta_k$, then the line minimizer is outside the trust region and we have that

$$\alpha_k^{\mathrm{N}} = \frac{\Delta_k}{\|s_k^Q\|_{M_k}}.$$

Finally, we consider the case where the curvature of the model along the Newton-like step is negative, that is, when $g_k^\top s_k^Q > 0$. In that case, the minimizer lies on the boundary of the trust region, and thus

$$\alpha_k^{\mathrm{N}} = -\frac{\Delta_k}{\|s_k^Q\|_{M_k}}.$$

By combining all cases, one concludes that

$$p_k^{\mathrm{N}} = \alpha_k^{\mathrm{N}} s_k^Q, \quad \text{where} \quad \alpha_k^{\mathrm{N}} = \min\left\{1, -\operatorname{sgn}(g_k^\top s_k^Q)\frac{\Delta_k}{\|s_k^Q\|_{M_k}}\right\}.$$

2. Suppose that the Newton-like step lies on the border of the trust region, i.e., $p_k^{\mathrm{N}} = \alpha_k^{\mathrm{N}} s_k^Q = -\operatorname{sgn}(g_k^\top s_k^Q)\frac{\Delta_k}{\|s_k^Q\|_{M_k}} s_k^Q$. The latter step is a stationary point of the subproblem (25) if and only if there exists a Lagrange multiplier $\lambda_k^{\mathrm{N}} \geq 0$ such that

$$(B_k + \lambda_k^{\mathrm{N}} M_k) p_k^{\mathrm{N}} = -g_k.$$

Substituting $p_k^{\mathrm{N}} = \alpha_k^{\mathrm{N}} s_k^Q$ in the latter equation, one has

$$\lambda_k^{\mathrm{N}} M_k s_k^Q = \left(1 - \frac{1}{\alpha_k^{\mathrm{N}}}\right) g_k. \tag{30}$$

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

$$\lambda_k^{\mathrm{N}} = \left(1 - \frac{1}{\alpha_k^{\mathrm{N}}}\right)\frac{g_k^\top s_k^Q}{\|s_k^Q\|_{M_k}^2} = \frac{g_k^\top s_k^Q}{\theta}\left(1 + \operatorname{sgn}(g_k^\top s_k^Q)\frac{\|s_k^Q\|_{M_k}}{\Delta_k}\right).$$

By replacing the value of λ_k^{N} in (30), we obtain that $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g_k$ where $\theta_k = \|s_k^Q\|_{M_k}^2 > 0$.

3. Indeed, when the step p_k^{N} lies on the boundary of the trust-region and $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g$. Then applying item (1) of Theorem 5.1, we see that

$$(B_k + \lambda_k^{\mathrm{N}} M_k) p_k^{\mathrm{N}} = -g_k$$

with $\lambda_k^{\mathrm{N}} = \frac{g_k^\top s_k^Q}{\theta_k} \left(1 + \mathrm{sgn}(g_k^\top s_k^Q) \frac{\|s_k^Q\|_{M_k}}{\Delta_k}\right) > 0$. Applying [8, Theorem 7.4.1], we see that if we assume that the matrix $B_k + \lambda_k^{\mathrm{N}} M_k$ is positive definite, then p_k^{N} is the unique minimizer of the subproblem (25). ■

Given an SPD matrix M_k that satisfies the secant equation $M_k s_k^Q = \frac{\theta_k}{g_k^\top s_k^Q} g_k$, item (3) of Theorem 5.1 states that the step p_k^{N} is the global minimizer of the associated subproblems over the subspace $\mathcal{L}_k$ as far as the matrix $B_k + \lambda_k^{\mathrm{N}} M_k$ is SPD. We note that λ_k^{N} goes to infinity as the trust region radius Δ_k goes to zero, meaning that the matrix $B_k + \lambda_k^{\mathrm{N}} M_k$ will be SPD as far as Δ_k is chosen to be sufficiently small. Since the TR update mechanism allows to shrink the value of Δ_k (when the iteration is declared as unsuccessful), satisfying the targeted condition will be geared automatically by the TR algorithm.

Again, when Assumption 3.1 holds, we note that *unsuccessful* iterations in the TR Algorithm require only updating the value of the TR radius Δ_k and the current step direction is kept unchanged. For such iterations, as far as there exists a matrix M_k such as $M_k s_k^Q = \frac{\beta_k \|s_k^Q\|^2}{g_k^\top s_k^Q} g_k$ where $\beta_k \in]\beta_{\min}, \beta_{\max}[$ and $\beta_{\max} > \beta_{\min} > 0$, the approximate solution of the TR subproblem is obtained only by updating the step-size α_k^{N}. This means that the computational cost of unsuccessful iterations do not requires solving any extra subproblem. We note that during the application of the algorithm, we will take θ_k of the form $\beta_k \|s_k^Q\|_2^2$, where $\beta_k \in]\beta_{\min}, \beta_{\max}[$ and $0 < \beta_{\min} < \beta_{\max}$. Such choice of the parameter θ_k ensures that the proposed M_k-norm uniformly equivalent to the l_2 one along the iterations (see Theorem 4.1).

In this setting, TR algorithms behaves as an LS method with a specific backtracking strategy. In fact, at the k^{th} iteration, the step is of the form $p_k = \alpha_k s_k^Q$ where s_k^Q is the (approximate) solution of the linear system $B_k s = -g_k$. The step length $\alpha_k > 0$ is chosen such as

$$\frac{f(x_k) - f(x_k + p_k)}{f(x_k) - m_k^Q(p_k)} \geq \eta \quad \text{and} \quad m_k^Q(p_k) \leq m_k^Q(-\alpha_k^{\mathrm{c}} g_k). \tag{31}$$

The values of α_k and α_k^{c} are computed respectively as follows:

$$\alpha_k \;=\; \min\left\{1, -\mathrm{sgn}(g_k^\top s_k^Q) \frac{\Delta_k}{\beta_k^{1/2} \|s_k^Q\|}\right\} \tag{32}$$

and

$$\alpha_k^{\mathrm{c}} = \begin{cases} \frac{\Delta_k}{\chi_k^{1/2} \|g_k\|} & \text{if } g_k^\top B_k g_k \leq 0 \text{ or } \frac{g_k^\top B_k g_k}{\|g_k\|} \geq \frac{\Delta_k}{\chi_k^{1/2}}, \\ \frac{g_k^\top B_k g_k}{\|g_k\|^2} & \text{else.} \end{cases} \tag{33}$$

where $\chi_k = \beta_k \left(\frac{5}{2} - \frac{3}{2}\cos(\varpi_k)^2 + 2\left(\frac{1-\cos(\varpi_k)^2}{\cos(\varpi_k)}\right)^2 \right)$ and $\cos(\varpi_k) = \frac{g_k^\top s_k^Q}{\|g_k\|\|s_k^Q\|}$. Δ_k is initially equals to the current value of the TR radius (as in the original TR algorithm). For large values of α_k the sufficient decrease condition (31) may not be satisfied, in this case, the value of Δ_k is contracted using the factor τ_1. Iteratively, the value of α_k is updated and the acceptance condition (31) is checked again until its satisfaction. Algorithm 5 details the adaptation of the classical TR algorithm when our proposed M_k-norm is used. We denote the final algorithm by LS-TR as it behaves as an LS algorithm.

Algorithm 5: LS-TR algorithm.

Data: select an initial point x_0 and the constants $0 < \eta < 1$, $0 < \epsilon_d \leq 1$, $0 \leq \tau_1 < 1 \leq \tau_2$, and $0 < \beta_{\min} < \beta_{\max}$. Set the initial TR radius $\Delta_0 > 0$ and $\Delta_{\max} > \Delta_0$.

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

  Choose a parameter $\beta_k \in]\beta_{\min}, \beta_{\max}[$;

  Let s_k^Q be an approximate stationary point of m_k^Q satisfying $|g_k^\top s_k^Q| \geq \epsilon_d \|g_k\| \|s_k^Q\|$;

  Set α_k and α_k^c using (32) and (33);

  while condition (31) is not satisfied **do**

    Set $\Delta_k \leftarrow \tau_1 \Delta_k$, and update α_k and α_k^c using (32) and (33);

  end

  Set $p_k = \alpha_k s_k^Q$, $x_{k+1} = x_k + p_k$ and $\Delta_{k+1} = \min\{\tau_2 \Delta_k, \Delta_{\max}\}$;

end

As far as the objective function f is continuously differentiable, its gradient is Lipschitz continuous, and its approximated Hessian B_k is bounded for all iterations, the TR algorithm is globally convergent and will required a number of iterations of order ϵ^{-2} to produce a point x_ϵ with $\|\nabla f(x_\epsilon)\| \leq \epsilon$ [14].

We note that the satisfaction of Assumption 3.1 is not problematic. As suggested for LS-ARC algorithm, one can modify the matrix B_k using regularization techniques (as far as the Hessian approximation is kept uniformly bounded from above all over the iterations, the global convergence and the complexity bounds will still hold [8]).

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 $s_k = \delta_k d_k$ where $d_k = s_k^Q$ if $-g_k^\top s_k^Q \geq \epsilon_d \|g_k\| \|s_k^Q\|$ (s_k^Q is being an approximate stationary point of m_k^Q) otherwise $d_k = -g_k$, and the step length $\delta_k > 0$ is chosen such as

$$f(x_k + s_k) \leq f(x_k) + \eta s_k^\top g_k, \tag{34}$$

where $\eta \in]0, 1[$. The appropriate value of δ_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 $\nabla^2 f(x_k)$, and $\epsilon_d = 10^{-3}$ for both

LS-ARC and LS-ARMIJO. Other numerical experiments (not reported here) with different values of ϵ_d (for instance $\epsilon_d = 10^{-1}$, 10^{-6}, and 10^{-12}) lead to almost the same results.

By way of comparison, we have also implemented the standard ARC/TR algorithms (see Algorithms 1 and 4) 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

$$\eta = 0.1,\ \nu_1 = 0.5,\ \nu_2 = 2,\ \sigma_0 = 1,\ \text{and } \sigma_{\min} = 10^{-16};$$

for GLTR-TR and LS-TR

$$\eta = 0.1,\ \tau_1 = 0.5,\ \tau_2 = 2,\ \Delta_0 = 1,\ \text{and } \Delta_{\max} = 10^{16};$$

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 β_k. A possible choice for β_k is $|g_k^\top s_k^Q|/\|s_k^Q\|^2$. This choice is inspired from the fact that, when the Hessian matrix B_k is SPD, this update corresponds to use the energy norm, meaning that the matrix M_k is set 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 β_k as follows: $\beta_k = 10^{-4}\sigma_k^{-2/3}$ 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 approximately the Newton step. Similarly, for the LS-TR method, we set $\beta_k = 1$ and this allows LS-TR to use the Newton step at the start of the backtracking strategy (as in LS-ARMIJO method).

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 problems with their default parameters. Regarding the algorithms LS-TR, LS-ARC, and LS-ARMIJO, we approximate the solution of the linear system $B_k s = -g_k$ using the MINRES Matlab solver. The latter method is a Krylov subspace method designed to solve symmetric linear systems [19]. 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, for LS-ARC/LS-TR , Assumption 3.1 was not violated frequently. The restoration of this assumption was ensured by performing iterations of GLRT-ARC/GLTR-TR (with the ℓ_2-norm) until a new successful iteration is found.

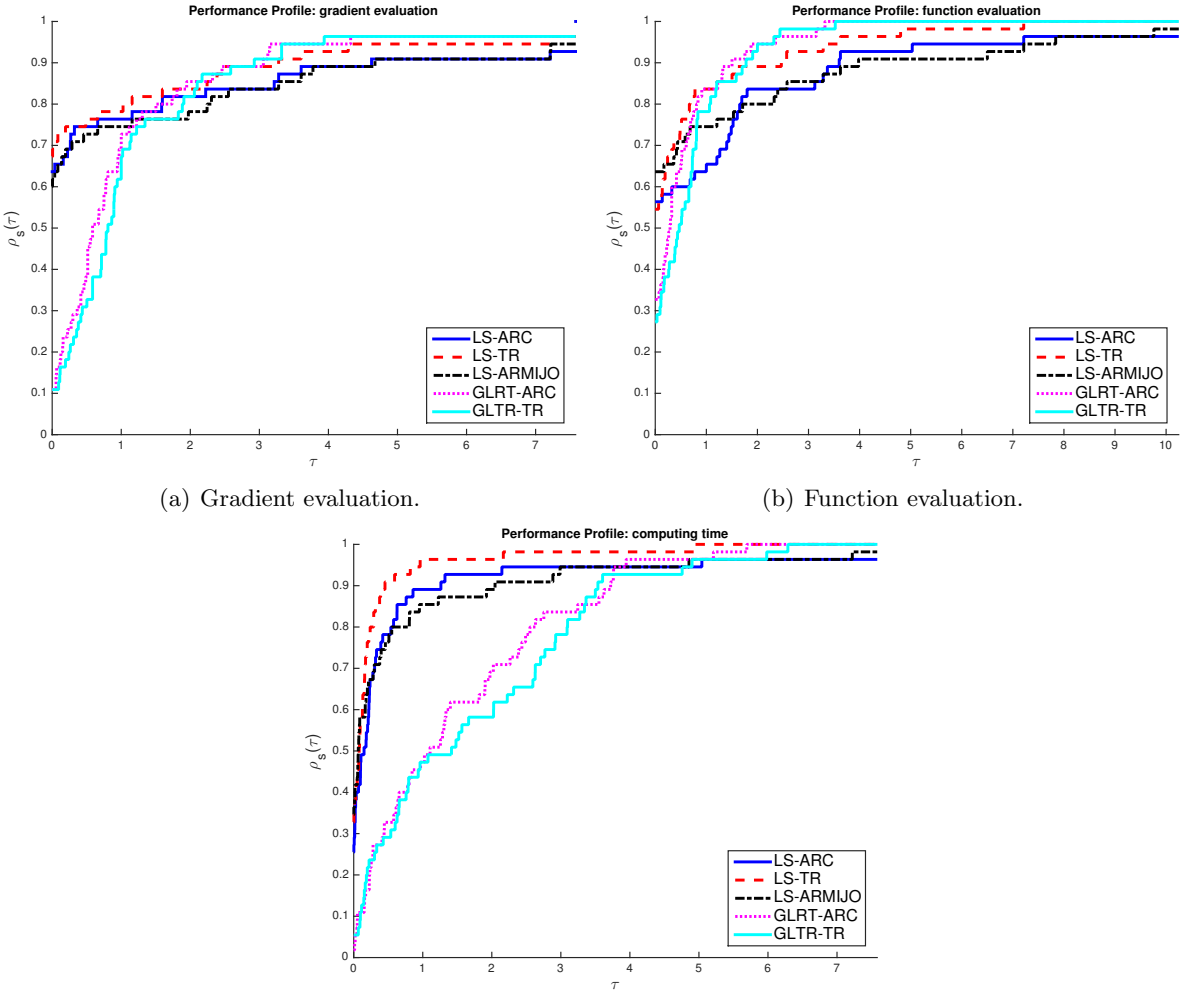
To compare the performance of the algorithms we use performance profiles proposed by Dolan and Moré [11] over a variety of problems. Given a set of problems $\mathcal{P}$ (of cardinality $|\mathcal{P}|$) and a set of solvers $\mathcal{S}$, the performance profile $\rho_s(\tau)$ of a solver s is defined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

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

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

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

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 τ 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 $\tau = 0$).



(a) Gradient evaluation.

(b) Function evaluation.

(c) CPU time.

Figure 1: Performance profiles for 62 large scale optimization problems (i.e., $1000 \leq n \leq 10000$).

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.

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

For the LS approaches, one can see that LS-TR displays better performance compared to LS-ARMIJO on the tested problems. The main difference between the two LS algorithms is the strategy of choosing the search direction whenever $g_k^\top s_k^Q > 0$. In our tested problems, the obtained performance using LS-TR suggests that going exactly in the opposite direction $-s_k^Q$, whenever s_k^Q is not a descent direction, can be seen as a good strategy compared to LS-ARMIJO.

7 Conclusions

In this paper, we have proposed the use of a specific norm in ARC/TR. With this norm choice, we have shown that the trial step of ARC/TR is getting collinear to the quasi-Newton direction. The obtained ARC/TR algorithm behaves as LS algorithms with a specific backtracking strategy. Under mild assumptions, the proposed scaled norm was shown to be uniformly equivalent to the Euclidean norm. In this case, the obtained LS algorithms enjoy the same convergence and complexity properties as ARC/TR. We have also proposed a second order version of the LS algorithm derived from ARC with an optimal worst-case complexity bound of order $\epsilon^{-3/2}$. 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 β_k on the performance of the proposed LS approaches. Also, the analysis of the second order version of ARC suggests that taking the Newton direction is suitable for defining a line-search method with an optimal worst-case complexity bound of order $\epsilon^{-3/2}$. It would be interesting to confirm the potential of the proposed line search strategy compared to the classical LS approaches using extensive numerical tests.

References

[1] E. Bergou, Y. Diouane, and S. Gratton. On the use of the energy norm in trust-region and adaptive cubic regularization subproblems. *Comput. Optim. Appl.*, 68(3):533–554, 2017.

[2] E. G. Birgin, J. L. Gardenghi, J. M. Martínez, S. A. Santos, and Ph. L. Toint. Worst-case evaluation complexity for unconstrained nonlinear optimization using high-order regularized models. *Math. Program.*, 163(1):359–368, 2017.

[3] E. G. Birgin and J. M. Martínez. The use of quadratic regularization with a cubic descent condition for unconstrained optimization. *SIAM J. Optim.*, 27(2):1049–1074, 2017.

[4] C. Cartis, N. I. M. Gould, and Ph. L. Toint. Adaptive cubic overestimation methods for unconstrained optimization. Part II: worst-case function- and derivative-evaluation complexity. *Math. Program.*, 130(2):295–319, 2011.

[5] C. Cartis, N. I. M. Gould, and Ph. L. Toint. Adaptive cubic regularisation methods for unconstrained optimization. Part I: Motivation, convergence and numerical results. *Math. Program.*, 127(2):245–295, 2011.

[6] C. Cartis, N. I. M. Gould, and Ph. L. Toint. Optimality of orders one to three and beyond: characterization and evaluation complexity in constrained nonconvex optimization. Technical report, 2017.

[7] C. Cartis, Ph. R. Sampaio, and Ph. L. Toint. Worst-case evaluation complexity of non-monotone gradient-related algorithms for unconstrained optimization. *Optimization*, 64(5):1349–1361, 2015.

[8] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*. SIAM, Philadelphia, PA, USA, 2000.

[9] F. E. Curtis, D. P. Robinson, and M. Samadi. A trust region algorithm with a worst-case iteration complexity of $O(\epsilon^{-3/2})$ for nonconvex optimization. *Math. Program.*, 162(1):1–32, 2017.

[10] J. E. Dennis and R. B. Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations*. Prentice-Hall Inc, Englewood Cliffs, NJ, 1983.

[11] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Math. Program.*, 91(2):201–213, 2002.

[12] N. I. M. Gould, D. Orban, and Ph. L. Toint. GALAHAD, a library of thread-safe fortran 90 packages for large-scale nonlinear optimization. *ACM Trans. Math. Softw.*, 29(4):353–372, 2003.

[13] N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEst: a constrained and unconstrained testing environment with safe threads for mathematical optimization. *Comput. Optim. Appl.*, 60(3):545–557, 2015.

[14] S. Gratton, A. Sartenaer, and Ph. L. Toint. Recursive trust-region methods for multiscale nonlinear optimization. *SIAM J. Optim.*, 19(1):414–444, 2008.

[15] A. Griewank. The modification of Newton's method for unconstrained optimization by bounding cubic terms. Technical Report NA/12, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, United Kingdom, 1981.

[16] J. M. Martínez and M. Raydan. Cubic-regularization counterpart of a variable-norm trust-region method for unconstrained minimization. *J. Global Optim.*, 68(2):367–385, 2017.

[17] Y. Nesterov. *Introductory Lectures on Convex Optimization*. Kluwer Academic Publishers, 2004.

[18] Y. Nesterov and B. T. Polyak. Cubic regularization of Newton's method and its global performance. *Math. Program.*, 108(1):177–205, 2006.

[19] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12(4):617–629, 1975.

[20] Y. Yuan. Recent advances in trust region algorithms. *Math. Program.*, 151(1):249–281, 2015.