On the use of the energy norm in trust-region and adaptive cubic regularization subproblems

E. Bergou * Y. Diouane† S. Gratton‡

March 6, 2017

Abstract

We consider solving unconstrained optimization problems by means of two popular globalization techniques: trust-region (TR) algorithms and adaptive regularized framework using cubics (ARC). Both techniques require the solution of a so-called "subproblem" in which a trial step is computed by solving an optimization problem involving an approximation of the objective function, called "the model". The latter is supposed to be adequate in a neighborhood of the current iterate. In this paper, we address an important practical question related with the choice of the norm for defining the neighborhood. More precisely, assuming here that the Hessian B of the model is symmetric positive definite, we propose the use of the so-called "energy norm" – defined by $\|x\|_B = \sqrt{x^T B x}$ for all $x \in \mathbb{R}^n$ – in both TR and ARC techniques. We show that the use of this norm induces remarkable relations between the trial step of both methods that can be used to obtain efficient practical algorithms. We furthermore consider the use of truncated Krylov subspace methods to obtain an approximate trial step for large scale optimization. Within the energy norm, we obtain line search algorithms along the Newton direction, with a special backtracking strategy and an acceptability condition in the spirit of TR/ARC methods. The new line search algorithm, derived by ARC, enjoys a worst-case iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$. We show the good potential of the energy norm on a set of numerical experiments.

Keywords: Nonlinear optimization, unconstrained optimization, trust-region algorithm, adaptive regularized framework using cubics, line search algorithm, energy norm, Krylov subspace methods, conjugate gradient.

1 Introduction

In this paper, we consider the following unconstrained optimization problem:

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

where $f : \mathbb{R}^n \to \mathbb{R}$ is a given smooth function. We propose to solve (1) by means of trust-region (TR) methods [7] or an adaptive regularized framework using cubics (ARC) [6]. In recent years,

*MaIAGE, INRA, Université Paris-Saclay, 78350 Jouy-en-Josas, France (el-houcine.bergou@jouy.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).

there has been a constant interest in the TR techniques [7, 11, 9], while ARC algorithms for unconstrained optimization enjoy a growing interest [3, 6, 5, 16, 14, 4]. The key principle of these two approaches is the (possibly iterative) minimization of objective function models (often, but not always, quadratic or cubic). During such minimization process, a control of the step length is used either trough an explicit constraint formulation or through a specific penalization term; that minimization stage is known as the "subproblem". Both TR and ARC frameworks can also be seen as regularization techniques as they control and impose restrictions on a certain parameter (a TR radius or a regularization parameter). TR algorithms are well established and studied, see [7] and references therein, whereas ARC methods are relatively newer and exhibit good performance in terms of global and local convergence properties as we briefly describe now. Numerical experiments indicate that ARC algorithms can be competitive with TR approaches when solving small-scale problems [6]. ARC algorithms are also famous since they improve substantially the worst-case iteration and gradient evaluation complexity over the classical TR methods [5]. Such improvement has been recently extended to modified TR methods [9] where a specific step update mechanism is proposed. The proposed TR algorithm exhibits the same worst-case complexity bound as ARC algorithms and enjoys the same convergence properties (fast local and global convergence) as the classical TR methods.

For both frameworks, the trial step from one iterate to the next is approximated by minimizing a given model of the objective function while the next iterate is maintained in a prespecified neighborhood of the current iterate. An important practical aspect of TR and ARC techniques is on the use of an appropriate description of the neighborhood. In this paper, we focus on the neighborhoods defined as spheres for some scaled versions of the Euclidean norm, i.e. $\|x\|_M = \sqrt{x^T M x}$ where M is a symmetric positive definite (SPD) matrix. Under such consideration, approximating the subproblem solution requires (approximately) solving a non-linear system, derived from the optimality conditions, by applying a zero-finding solver to the so-called "secular equation" [7]. Practical approaches to get an approximate solution are proposed in [7, 4, 6, 11], where the solution of the secular equation is typically approximated over specific evolving subspaces using Krylov methods. The main drawback of such approaches is there expensive computational cost as they may require solving multiple linear systems in sequence.

Our approach for choosing M is related to ideas developed, for instance, when solving infinite-dimensional linear problems arising in elliptic partial differential equations. In fact, it is common to recourse to iterative techniques for solving the corresponding linear systems after discretization. In this setting, finding a sound stopping criterion is essential to overcome both over- and under- solving the regarded problem, i.e. appropriately balance the error due to the truncation of the iterative solver and the discretization error. It can be shown that it is theoretically natural and practically relevant to work with the scaled Euclidean norm where M is chosen as the matrix of the linear system itself [2, 1]. For optimization problems, the use of the absolute-value of the Hessian (that corresponds to the energy norm in the case of a positive definite Hessian) to define the trust region in the TR algorithms was proposed in [7, Section 7.7.1] as "the ideal trust region" that reflects the proper scaling of the underlying problem. For a large scale indefinite Hessian, computing the absolute-value is certainly a computationally expensive task. This means that for large scale optimization problems the use of the absolute-value energy norm can be seen as out of reach, and was not further considered in [7]. In this paper, we investigate the use of the energy norm instead of a general scaled norm for both the TR and ARC algorithms in the situation where the Hessian of the model is SPD, which we believe is common. To mention

only one widespread application, it occurs when a Gauss-Newton approach is used for solving nonlinear least-squares problems.

In this paper we show that, for locally convex model in TR/ARC framework, the computational cost of the associated subproblem is nearly the same as solving a linear system (i.e., approximating a minimizer of the associated quadratic model), and does not involve nonlinear iterations. 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. In this case, we propose a truncated Krylov subspace method for the subproblem. Using our subproblem solvers, we also consider the performance of the overall unconstrained optimization solver and show that the energy norm may produce much better results than a standard Euclidean norm based algorithm. Most notably, our approach is very economical because the dominant computational cost of our TR/ARC algorithms is mainly the cost of successful iterations in the TR/ARC algorithms, solving the subproblem for unsuccessful iterations being straight-forward and inexpensive. The proposed approaches are in practice line search algorithms along the Newton direction, with a special backtracking strategy and an acceptability condition in the spirit of TR/ARC methods. We show that the proposed line search approach, derived by the ARC framework, enjoys a worst-case iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$.

The outline of this paper is as follows. In Section 2 we recall a unified framework for unconstrained optimization gathering both TR algorithms and the ARC approaches and assuming a general scaled norm is used for the step size control. In Section 3, we present closed formulas for the solution of the subproblem when the energy norm is considered, and propose both direct and iterative scheme for computing it. In Section 4 we derive the overall unconstrained optimization algorithm and discuss possible stopping criteria to maintain the complexity of TR and ARC when the subproblem is solved iteratively. In Section 5, preliminary numerical experiments with basic implementations are presented that show the good behavior of our energy norm algorithm. Finally, in Section 6 we draw some perspectives and conclusions.

2 TR/ARC algorithms

In this section, we describe the subproblem that occurs in the k^{th} iteration when solving the optimization problem (1) in both the TR algorithms (e.g., [7]) and ARC algorithms (e.g., [6]). Formally in a basic TR algorithm, one computes a trial step s_k^{TR} by approximately solving

$$\begin{aligned} \min_{s\in\mathbb{R}^n} \quad & f(x_k) + s^T g_k + \tfrac{1}{2} s^T B_k s \quad = m_k^Q(s) \\ \text{s. t.} \quad & \|s\|_{M_k} \leq \Delta_k, \end{aligned} \tag{2}$$

where $\Delta_k > 0$ is known as the TR radius, $g_k = \nabla f(x_k)$ is the gradient of f at the current iterate x_k, and B_k is a symmetric approximation to the local Hessian of f at the point x_k. The scaled norm $\|.\|_{M_k}$ may vary along the iterations and M_k is a symmetric positive definite matrix.

Once the trial step s_k^{TR} is determined, the objective function is computed at $x_k + s_k^{\text{TR}}$ and compared with the value predicted by the model at this point. If the model value predicts sufficiently well the objective function, the trial point $x_k + s_k^{\text{TR}}$ will be accepted and the TR is eventually expanded. If the model turns out to predict poorly the objective function, the trial point is rejected and the TR is contracted. For a basic ARC algorithm, the trial step s_k^{ARC} is

computed as an approximate minimizer of a cubic model m_k^C:

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

where $\sigma_k > 0$ is a dynamic positive parameter that more or less plays the same role as the inverse of the TR radius (see [6]). Similarly to TR algorithms, the update of the parameter σ_k takes into account the agreement between the objective function f and the model m_k^C. The parameter σ_k and the current iterate are updated following similar principles as those used for Δ_k and x_k in the TR methods. More formally, for both frameworks, a ratio between the actual reduction and the predicted reduction plays an important role to decide whether the trial step is acceptable or not. In the TR case, the ratio is of the following form:

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

and in the adaptive cubic regularization case is of the form

$$\rho_k^{\rm ARC} \;=\; \frac{f(x_k) - f(x_k + s_k^{\rm ARC})}{f(x_k) - m_k^C(s_k^{\rm ARC})}. \tag{5}$$

To keep the notation simple, and unless it introduces some ambiguity, we will use ρ_k for both $\rho_k^{\rm TR}$ and $\rho_k^{\rm ARC}$. For a given scalars $0 < \eta_1 \leq \eta_2 < 1$, the k^{th} iteration will be said *very successful* if $\rho_k \geq \eta_2$, *successful* if $\rho_k \in [\eta_1, \eta_2)$, and *unsuccessful* otherwise. In all *successful* iterations we set $x_{k+1} = x_k + s_k$; otherwise the current iterate is kept unchanged $x_{k+1} = x_k$. Algorithm 1 gives a detailed description of both algorithms (TR and ARC frameworks). The generic name `'ALGO'` stands either for `'TR'` or `'ARC'` in the sequel.

For the purpose of this paper, we will assume that the approximate Hessian matrix B_k is positive definite for all iterations.

Assumption 2.1 *For all k, the approximate Hessian matrix B_k of the model is symmetric positive definite.*

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, in the TR case, is defined as follows:

$$s_{k,\rm Cauchy}^{\rm TR} = -\alpha_{k,\rm Cauchy}^{\rm TR} g_k, \quad \text{where} \quad \alpha_{k,\rm Cauchy}^{\rm TR} = \arg\min_{t>0;\|x_k - tg_k\|_{M_k}\leq\Delta_k} m_k^Q(-tg_k), \tag{6}$$

and in the adaptive cubic regularization case:

$$s_{k,\rm Cauchy}^{\rm ARC} = -\alpha_{k,\rm Cauchy}^{\rm ARC} g_k, \quad \text{where} \quad \alpha_{k,\rm Cauchy}^{\rm ARC} = \arg\min_{t>0} m_k^C(-tg_k). \tag{7}$$

The Cauchy step associated to the minimization of the unconstrained quadratic model $m^Q(s)$ is of the following form:

$$s_{k,\rm Cauchy}^{Q} = -\alpha_{k,\rm Cauchy}^{Q} g_k, \quad \text{where} \quad \alpha_{k,\rm Cauchy}^{Q} = \arg\min_{t>0} m_k^Q(-tg_k) = \frac{\|g_k\|^2}{\|g_k\|_{B_k}^2}. \tag{8}$$

Under classical assumptions, convergence results of Algorithm 1 can be found in [6, 7]. In the rest of the paper, we will mostly focus on the subproblem minimization for a given iteration k. The iteration subscript k will therefore be dropped to keep the notations simple.

Algorithm 1: A generic TR/ARC algorithm.

Initialization: Select an initial point x_0 and constants $0 < \eta_1 < \eta_2 < 1$. If `ALGO='TR'`, set the initial radius Δ_0 and the constants $0 \leq \tau_1 \leq \tau_2 \leq 1$. Else set the initial regularization $\sigma_0 > 0$ and the constants $1 < \nu_1 \leq \nu_2$. Set $k = 0$.

Until convergence:

1. Model definition :

Compute the model gradient g_k and its Hessian B_k.

2. Compute the step :

If `ALGO='TR'`, compute the step s_k as an approximate solution of (2). Else compute the step s_k as an approximate solution of (3).

3. Accept the trial point :

If `ALGO='TR'`, compute $\rho_k = \rho_k^{\text{TR}}$ as in (4). Else, compute $\rho_k = \rho_k^{\text{ARC}}$ as in (5).

If $\rho_k \geq \eta_1$, set $x_{k+1} = x_k + s_k$; otherwise $x_{k+1} = x_k$.

4. Parameter update:

If `ALGO='TR'` set

$$\Delta_{k+1} \in \begin{cases} [\Delta_k, +\infty) & \text{if} \quad \rho_k \geq \eta_2, \quad [\textit{very successful}] \\ [\tau_2 \Delta_k, \Delta_k] & \text{if} \quad \eta_1 \leq \rho_k < \eta_2, \quad [\textit{successful}] \\ [\tau_1 \Delta_k, \tau_2 \Delta_k] & \text{Otherwise.} \quad [\textit{unsuccessful}] \end{cases}$$

Else set

$$\sigma_{k+1} \in \begin{cases} (0, \sigma_k] & \text{if} \quad \rho_k \geq \eta_2, \quad [\textit{very successful}] \\ [\sigma_k, \nu_1 \sigma_k] & \text{if} \quad \eta_1 \leq \rho_k < \eta_2, \quad [\textit{successful}] \\ [\nu_1 \sigma_k, \nu_2 \sigma_k] & \text{Otherwise.} \quad [\textit{unsuccessful}] \end{cases}$$

Increment k by one.

3 TR/ARC subproblems with the energy norm

In the following, we suppose that Assumption 2.1 holds and that the matrix M in the scaled norm is equal to the model Hessian matrix B. In this section we derive the formula for computing the subproblem solution of the TR/ARC algorithms when the energy norm is used.

Each iteration of Algorithm 1 requires mainly to solve (possibly inexactly) a subproblem of type (2)-(3). Finding an efficient solver to the subproblem is obviously essential to keep the work-per-iteration low. We prove that in the energy norm setting, approximating the solutions of (2)-(3) is as expensive as approximating the solution of a single unconstrained quadratic optimization problem.

3.1 Exact solution of the TR/ARC subproblem

In what follows, we will denote by s_*^Q the exact solution of the unconstrained quadratic optimization problem:

$$s_*^Q = \arg\min_{s\in\mathbb{R}^n} m^Q(s) \quad = \quad -B^{-1}g. \tag{9}$$

Theorem 3.1 *The exact solutions $s_*^{\rm TR}$ and $s_*^{\rm ARC}$ of the subproblems (2)-(3) are respectively of the following form:*

$$s_*^{\rm TR} \quad = \quad \delta_*^{\rm TR} s_*^Q, \quad \textit{where} \quad \delta_*^{\rm TR} = \min\left(1, \frac{\Delta}{\|s_*^Q\|_B}\right), \tag{10}$$

and

$$s_*^{\rm ARC} \quad = \quad \delta_*^{\rm ARC} s_*^Q, \quad \textit{where} \quad \delta_*^{\rm ARC} = \frac{2}{1+\sqrt{1+4\sigma\|s_*^Q\|_B}}. \tag{11}$$

Proof. Let $s_*^{\rm TR}$ be a solution of (2). The matrix B is positive definite, thus if $\|s_*^Q\|_B \leq \Delta$, then the optimization problem (2) has a unique solution $s_*^{\rm TR} = s_*^Q$. The case where $\|s_*^Q\|_B > \Delta$ implies that the solution $s_*^{\rm TR}$ lies on the boundary of the TR (i.e. $\|s_*^{\rm TR}\|_B = \Delta$). It follows from the first and second-order necessary and sufficient (as B is positive definite) optimality conditions on the subproblem (2), that $s_*^{\rm TR}$ is unique and satisfies

$$g + Bs_*^{\rm TR} + \lambda_*^{\rm TR} Bs_*^{\rm TR} \quad = \quad 0,$$

with $\lambda_*^{\rm TR} \geq 0$ is the so-called Lagrange multiplier. Which implies that

$$s_*^{\rm TR} \quad = \quad \frac{1}{1+\lambda_*^{\rm TR}} s_*^Q.$$

Since $\|s_*^{\rm TR}\|_B = \Delta$, one concludes that

$$\lambda_*^{\rm TR} = \frac{\|s_*^Q\|_B}{\Delta} - 1 \quad \text{and } s_*^{\rm TR} = \frac{\Delta}{\|s_*^Q\|_B} s_*^Q.$$

On the other hand, let $s_*^{\rm ARC}$ be a solution of (3). It follows from the first and second-order necessary optimality conditions at $s_*^{\rm ARC}$:

$$\nabla_s m^C(s_*^{\rm ARC}) \quad = \quad g + (1+\lambda_*^{\rm ARC})Bs_*^{\rm ARC} = 0,$$

and for all vectors $w \in \mathbb{R}^n$,

$$w^T \left(\nabla_{ss} m^C(s_*^{\rm ARC})\right) w \quad = \quad w^T \left((1+\lambda_*^{\rm ARC})B + \lambda_*^{\rm ARC}\left(\frac{Bs_*^{\rm ARC}}{\|s_*^{\rm ARC}\|_B}\right)\left(\frac{Bs_*^{\rm ARC}}{\|s_*^{\rm ARC}\|_B}\right)^T\right) w \geq 0.$$

where $\lambda_*^{\rm ARC} = \sigma\|s_*^{\rm ARC}\|_B$.

The matrix B is definite positive, hence for all $w \neq 0$

$$w^T \left((1+\lambda_*^{\text{ARC}})B + \lambda_*^{\text{ARC}} \left(\frac{Bs_*^{\text{ARC}}}{\|s_*^{\text{ARC}}\|_B} \right) \left(\frac{Bs_*^{\text{ARC}}}{\|s_*^{\text{ARC}}\|_B} \right)^T \right) w \geq w^T \left((1+\lambda_*^{\text{ARC}})B \right) w > 0,$$

thus s_*^{ARC} is the unique minimizer of (3) and of the following form

$$s_*^{\text{ARC}} \quad = \quad \frac{1}{1+\lambda_*^{\text{ARC}}} s_*^Q \quad \text{where } \lambda_*^{\text{ARC}} = \sigma \|s_*^{\text{ARC}}\|_B = \frac{\sigma}{1+\lambda_*^{\text{ARC}}} \|s_*^Q\|_B.$$

Then λ_* is solution of the following equation

$$\left(\lambda_*^{\text{ARC}}\right)^2 + \lambda_*^{\text{ARC}} - \sigma \|s_*^Q\|_B = 0.$$

Since $\lambda_*^{\text{ARC}} \geq 0$, one has

$$\lambda_*^{\text{ARC}} = \frac{-1 + \sqrt{1 + 4\sigma \|s_*^Q\|_B}}{2},$$

and hence, $s_*^{\text{ARC}} = \delta_*^{\text{ARC}} s_*^Q$, where $\delta_*^{\text{ARC}} = \frac{1}{1+\lambda_*^{\text{ARC}}} = \frac{2}{1+\sqrt{1+4\sigma\|s_*^Q\|_B}}$. ■

A direct consequence of this result is that solving exactly the TR or ARC subproblem (in the energy norm setting) is as expensive as minimizing the model m^Q without constraints. It simply amounts to scaling the solution of the linear system $Bs = -g$ by a scalar that is easy to evaluate. This means that solving the subproblem for any values of Δ or σ 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 of the subproblem using Krylov subspace methods

Theorem 3.1 shows that the dominant computational cost to get the exact solution of the subproblem is that of the computation of s_*^Q. For large scale optimization problems, computing s_*^Q can be prohibitively computationally expensive. We relax this requirement by letting the step s_*^Q be an approximation of the exact solution.

As far as the global convergence of Algorithm 1 is concerned, all what we need is that the solution of the subproblems (2)-(3) yields a decrease in the model (quadratic or cubic) that is as good as a fraction of the Cauchy decrease. The latter is obtained by minimizing the model along the negative gradient direction [6, 7]. 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 computation. This idea is, for instance, implemented in the standard Steihaug-Toint conjugate gradient algorithm for the TR method [7], in which a Cauchy step is computed and the model is further decreased by projection onto a sequence embedded Krylov subspaces. We now show how to use a similar idea to compute a solution of the subproblem in the energy norm that is cheap to compute and yields a global convergence of the minimization algorithm.

The truncated CG method belongs to the class of methods relying on Krylov subspaces, which are spanned by the current gradient g and by vectors formed by repeated multiplication of g by the system matrix B. Formally, the i^{th} Krylov subspace defined by B and g is defined by:

$$\mathcal{K}_i(B, g) = \text{span}\{g, Bg, \ldots, B^{i-1}g\}.$$

The short-hand notation $\mathcal{K}_i$ will be used in the sequel since the dependence on B and g is clear from the context. Note that these subspaces are nested, i.e., $\mathcal{K}_i \subset \mathcal{K}_{i+1}$.

Let A_i denote any matrix whose columns form a basis of $\mathcal{K}_i$. We will denote by s_i^Q the exact solution of the unconstrained quadratic optimization problem over the subspace $\mathcal{K}_i$:

$$s_i^Q = \arg\min_{s \in \mathcal{K}_i} m^Q(s) \quad = \quad -A_i(A_i^T B A_i)^{-1} A_i^T g. \tag{12}$$

The Cauchy step can be regarded as the first iteration of a Krylov subspace method when applied to the unconstrained quadratic model m^Q. Since the Krylov subspaces are nested, the decrease attained on the quadratic model at the first iteration of the subspace method (i.e., the Cauchy step) is kept along the iterations [17]:

$$m^Q(s_{i+1}^Q) \leq m^Q(s_i^Q) \leq m^Q(s_{\text{Cauchy}}^Q), \ i = 1, \ldots. \tag{13}$$

In Theorem 3.1, we showed that the exact solutions of the subproblems (2)-(3) are collinear to the step of the unconstrained quadratic subproblem. We now show that explicit formula can also be derived when using projection on Krylov subspace and that the TR and ARC projected solutions enjoy similar collinearity properties as in Theorem 3.1.

Theorem 3.2 *The exact solutions s_i^{TR} and s_i^{ARC} of the subproblems (2)-(3) over the $i-th$ Krylov subspace $\mathcal{K}_i$ are respectively of the following form:*

$$s_i^{\text{TR}} \quad = \quad \delta_i^{\text{TR}} s_i^Q, \quad \textit{where} \quad \delta_i^{\text{TR}} = \min\left(1, \frac{\Delta}{\|s_i^Q\|_B}\right), \tag{14}$$

and

$$s_i^{\text{ARC}} \quad = \quad \delta_i^{\text{ARC}} s_i^Q, \quad \textit{where} \quad \delta_i^{\text{ARC}} = \frac{2}{1 + \sqrt{1 + 4\sigma\|s_i^Q\|_B}}. \tag{15}$$

Proof. Let l denote the dimension of the subspace $\mathcal{K}_i$ and A_i be an $n \times l$ matrix whose columns form a basis of $\mathcal{K}_i$. Thus for all $s \in \mathcal{K}_i$, we have $s = A_i z$, for some $z \in \mathbb{R}^l$.

For the TR algorithm, let s_i^{TR} be the exact solution of the subproblem (2) over the subspace $\mathcal{K}_i$, and letting

$$s_i^{\text{TR}} \quad = \quad A_i z_*^{\text{TR}}, \tag{16}$$

we have that z_*^{TR} is the exact solution of the following optimization problem

$$\min_{\|z\|_{A_i^T M A_i} \leq \Delta} f(x) + z^T A_i^T g + \frac{1}{2} z^T A_i^T B A_i z. \tag{17}$$

Hence, applying Theorem 3.1 to the reduced minimization problem (17), it follows that

$$\arg \min_{\|z\|_{A_i^T BA_i} \leq \Delta} f(x) + z^T A_i^T g + \frac{1}{2} z^T A_i^T BA_i z \quad = \quad \min \left(1, \frac{\Delta}{\|z_*^Q\|_{A_i^T BA_i}} \right) z_*^Q$$

where z_*^Q is the exact solution of the following unconstrained quadratic optimization problem :

$$z_*^Q = \arg \min_{z \in \mathbb{R}^l} f(x) + z^T A_i^T g + \frac{1}{2} z^T A_i^T BA_i z \quad = \quad -(A_i^T BA_i)^{-1} A_i^T g. \tag{18}$$

Thus, from equation (16), one concludes that

$$\begin{aligned} s_i^{\mathrm{TR}} &= A_i \left(\min \left(1, \frac{\Delta}{\|z_*^Q\|_{A_i^T BA_i}} \right) z_*^Q \right) \\ &= \min \left(1, \frac{\Delta}{\|A_i z_*^Q\|_B} \right) A_i z_*^Q \\ &= \min \left(1, \frac{\Delta}{\|s_i^Q\|_B} \right) s_i^Q. \end{aligned}$$

In the other hand, for the ARC algorithm, let s_i^{ARC} be the exact solution of the subproblem (3) over the subspace $\mathcal{K}_i$, and letting

$$s_i^{\mathrm{ARC}} \quad = \quad A_i z_*^{\mathrm{ARC}}, \tag{19}$$

we have that z_*^{ARC} is the exact solution of the following optimization problem

$$\min_{z \in \mathbb{R}^l} f(x) + z^T A_i^T g + \frac{1}{2} z^T A_i^T BA_i z + \frac{1}{3} \sigma \|z\|^3_{A_i^T BA_i} \tag{20}$$

Hence, applying Theorem 3.1 to the reduced minimization problem (20), it follows that

$$\arg \min_{z \in \mathbb{R}^l} f(x) + z^T A_i^T g + \frac{1}{2} z^T A_i^T BA_i z + \frac{1}{3} \sigma \|z\|^3_{A_i^T BA_i} \quad = \quad \frac{2}{1 + \sqrt{1 + 4\sigma \|z_*^Q\|_{A_i^T BA_i}}} z_*^Q$$

Thus, from equation (19), one concludes that

$$\begin{aligned} s_i^{\mathrm{ARC}} &= A_i \left(\frac{2}{1 + \sqrt{1 + 4\sigma \|z_*^Q\|_{A_i^T BA_i}}} z_*^Q \right) \\ &= \frac{2}{1 + \sqrt{1 + 4\sigma \|A_i z_*^Q\|_B}} A_i z_*^Q \\ &= \frac{2}{1 + \sqrt{1 + 4\sigma \|s_i^Q\|_B}} s_i^Q. \end{aligned}$$

■

Since the first iteration of the Krylov subspace method when applied to the unconstrained quadratic $m^Q(s)$ corresponds exactly to the Cauchy point (i.e., $s_1^Q = s_{\mathrm{Cauchy}}^Q$). Theorem 3.2 implies that s_1^{TR} and s_1^{ARC} correspond respectively to the Cauchy steps $s_{\mathrm{Cauchy}}^{\mathrm{TR}}$ and $s_{\mathrm{Cauchy}}^{\mathrm{ARC}}$.

Corollary 3.1 *The Cauchy steps* $s^{\rm TR}_{\rm Cauchy}$ *and* $s^{\rm ARC}_{\rm Cauchy}$ *can be obtained from* $s^{Q}_{\rm Cauchy} = -\frac{\|g\|^2}{\|g\|^2_B} g$ *as follows:*

$$s^{\rm TR}_{\rm Cauchy} \;=\; \delta^{\rm TR}_{\rm Cauchy} s^{Q}_{\rm Cauchy}, \quad \textit{where} \quad \delta^{\rm TR}_{\rm Cauchy} = \min\left(1, \frac{\Delta}{\|s^{Q}_{\rm Cauchy}\|_B}\right), \tag{21}$$

and

$$s^{\rm ARC}_{\rm Cauchy} \;=\; \delta^{\rm ARC}_{\rm Cauchy} s^{Q}_{\rm Cauchy}, \quad \textit{where} \quad \delta^{\rm ARC}_{\rm Cauchy} = \frac{2}{1+\sqrt{1+4\sigma\|s^{Q}_{\rm Cauchy}\|_B}}. \tag{22}$$

4 TR/ARC algorithms using the energy norm

In [7, 6], the convergence of the TR/ARC algorithms is established in the case where the norms that are used to control the step size are iteration dependent and uniformly equivalent. In practice, the most common choices for these norms are the l_1, l_2, or l_∞ norms or scaled variant of those. For a general scaled norm (different from the energy norm) Newton or secant methods are used to solve the subproblem which require to solve iteratively many linear systems [6, 7].

When using the energy norm, notice that *unsuccessful* iterations of Algorithm 1 require only a parameter update (Δ if the TR method is used and σ otherwise) and the current step direction is kept unchanged. In this case, the approximate solutions of the subproblems are obtained only by updating the step-sizes $\delta^{\rm TR}$ and $\delta^{\rm ARC}$, which just involve one additional energy norm computation. This means that the computational cost of unsuccessful iterations is getting limited (see Theorems 3.1-3.2) compared to solving linear system as required in, say, the l_2 norm approach. As a consequence, the use of the energy norm in Algorithm 1 leads to a new algorithm where the dominant computational cost is mainly the cost of successful iterations. Algorithm 2 details the adaptation of the classical algorithms ARC and TR when the energy norm is used. The obtained algorithms are in practice line search algorithms along the direction s^Q_k with a special backtracking strategy and an acceptability condition of the form $\rho_k \geq \eta_1$, instead of the standard Armijo rule. We denote the new TR/ARC algorithm by TR-EN/ARC-EN algorithm, where EN is standing for the Energy Norm. Again the string `ALGO` denotes the name of the algorithm, i.e., it is either `'TR-EN'` or `'ARC-EN'`.

Based on the references [5, 6], the TR-EN/ARC-EN algorithm ensures convergence to first-order critical points and with steepest descent like function evaluation complexity bound of order ϵ^{-2} to guarantee

$$\|g\|_M \;\leq\; \epsilon \tag{23}$$

where $\epsilon > 0$ is pre-specified constant and M is a given symmetric positive definite matrix.

By imposing a more stringent termination condition on the solution of the trial step $s^{\rm ARC}$, one can improve function-evaluation complexity to be of the order of $\epsilon^{-3/2}$ to ensure (23) for the ARC-EN algorithm (see [5]). Such termination condition, known as the "s-rule", is of the type

$$\|\nabla m^C(s^{\rm ARC})\|_M \;\leq\; \kappa_{\rm S}\|s^{\rm ARC}\|^2_M \tag{24}$$

Algorithm 2: A generic TR-EN/ARC-EN algorithm.

Initialization: Select an initial point x_0 as well as constants $0 < \eta_1 < \eta_2 < 1$. If `ALGO='TR-EN'`, set the initial radius Δ_0 and the constants $0 \leq \tau_1 \leq \tau_2 \leq 1$. Else set the initial regularization $\sigma_0 > 0$ and the constants $01 < \nu_1 \leq \nu_2$. Set $k = 0$.

Until convergence:

1. Model definition :

Compute the gradient model g_k and its Hessian B_k.

2. Compute an approximated quadratic step :

Compute a step s_k^Q for which

$$m_k^Q(s_k^Q) \leq m_k^Q(s_{k,\text{Cauchy}}^Q),$$

where the Cauchy point

$$s_{k,\text{Cauchy}}^Q = -\alpha_{k,\text{Cauchy}}^Q g_k, \quad \text{and} \quad \alpha_{k,\text{Cauchy}}^Q = \arg\min_{t>0} m_k^Q(-tg_k) = \frac{\|g_k\|^2}{\|g_k\|_{B_k}^2}.$$

3. Until the iteration is successful

(i) Compute TR-EN/ARC-EN step:

If `ALGO='TR-EN'`, set $\delta_k = \min(1, \frac{\Delta_k}{\|s_k^Q\|_{B_k}})$. Else $\delta_k = \frac{2}{1+\sqrt{1+4\sigma_k\|s_k^Q\|_{B_k}}}$. Set $s_k = \delta_k s_k^Q$.

(ii) Accept the trial point:

If `ALGO='TR-EN'`, compute $\rho_k = \rho_k^{\text{TR}}$ as in (4). Else, compute $\rho_k = \rho_k^{\text{ARC}}$ as in (5). If $\rho_k \geq \eta_1$, set $x_{k+1} = x_k + s_k$ and the iteration is successful; otherwise $x_{k+1} = x_k$, $B_{k+1} = B_k$, $g_{k+1} = g_k$ and $s_{k+1}^Q = s_k^Q$.

(iii) Parameter update:

If `ALGO='TR-EN'` set

$$\Delta_{k+1} \in \begin{cases} [\Delta_k, +\infty) & \text{if} \quad \rho_k \geq \eta_2, \\ [\tau_2\Delta_k, \Delta_k] & \text{if} \quad \eta_1 \leq \rho_k < \eta_2, \\ [\tau_1\Delta_k, \tau_2\Delta_k] & \text{Otherwise.} \end{cases}$$

Else set

$$\sigma_{k+1} \in \begin{cases} (0, \sigma_k] & \text{if} \quad \rho_k \geq \eta_2, \\ [\sigma_k, \nu_1\sigma_k] & \text{if} \quad \eta_1 \leq \rho_k < \eta_2, \\ [\nu_1\sigma_k, \nu_2\sigma_k] & \text{Otherwise.} \end{cases}$$

Increment k by one.

for some constant $\kappa_{\rm S} > 0$ chosen at the start of the algorithm.

Using the energy norm, in the next proposition, we show that the "s-rule" condition (24) can be expressed only in terms of s^Q and ∇m^Q.

Proposition 4.1 *For ARC-EN algorithm, imposing the "s-rule" condition (24) is equivalent to the following condition*

$$\|\nabla m^Q(s^Q)\|_M \quad \leq \quad \kappa_s \delta^2 \|s^Q\|_M^2 = 4\kappa_s \left(\frac{\|s^Q\|_M}{1 + \sqrt{1 + 4\sigma \|s^Q\|_B}} \right)^2 \tag{25}$$

Proof. Indeed, one has $s^{\rm ARC} = \delta^{\rm ARC} s^Q$ thus

$$\begin{aligned} \nabla m^C(s^{\rm ARC}) &= g + B s^{\rm ARC} + \sigma \|s^{\rm ARC}\|_B B s^{\rm ARC} \\ &= g + \left(\delta^{\rm ARC} + \sigma (\delta^{\rm ARC})^2 \|s^Q\|_B \right) B s^Q \end{aligned}$$

By definition of $\delta^{\rm ARC}$, one has $\sigma \|s^Q\|_B (\delta^{\rm ARC})^2 + \delta^{\rm ARC} = 1$, which implies

$$\nabla m^C(s^{\rm ARC}) \quad = \quad g + B s^Q = \nabla m^Q(s^Q).$$

Hence, the "s-rule" condition (24) is being equivalent to the termination condition (25). ■

First order optimality conditions imply that $\|\nabla m^Q(s_*^Q)\|_M = 0$; the termination condition (25) is therefore satisfied when s^Q is the exact solution of the quadratic minimization problem. One hopes of course that the termination condition will occur well before convergence to the exact solution. Algorithm 3 summarized a second-order variant of ARC-EN which is guaranteed to have improved function-evaluation worst-case complexity of order $\epsilon^{-3/2}$ to ensure (23). Using Proposition 4.1, the proof is exactly the same as in [5]. In other words, we showed a possible way to derive a line search approach (i.e., ARC-EN$_{\rm (S)}$) enjoying a worst-case iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$.

Finally and most importantly, using the energy norm still leaves 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 (24) is met. This is expected to happen early along the Krylov iterations when the preconditioner for the linear system $Bs = -g$ is good enough. The next section investigates how TR-EN/ARC-EN algorithms perform in practice.

5 Numerical results

In this section we report the results of some preliminary experiments performed in order to assess the efficiency and the robustness of the proposed algorithms (`TR-EN` and `ARC-EN`). At the end of this section we report results of our proposed algorithms when different methods for approximating the solution of the subproblem are used. We implement all the algorithms as Matlab m-files.

5.1 Implementation details and test strategy

For the numerical experiments in this section, we solve exactly the subproblem (9) using the backslash operator (which uses the `LAPACK` Fortran library).

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

In each iteration k of Algorithm 2, compute s_k^Q in Step 2 as an approximate solution of

$$B_k s = -g_k,$$

such that the termination condition

$$\|B_k s_k^Q + g_k\|_M \quad \leq \quad \kappa_S \delta_k^2 \|s_k^Q\|_M^2 \tag{26}$$

is satisfied, where $\delta_k = \frac{2}{1+\sqrt{1+4\sigma_k \|s_k^Q\|_{B_k}}}$ and $\kappa_S > 0$ is a contant chosen at the beginning of the algorithm.

Since our proposed approach behaves in practice as line search procedures, in all our tests, we add a comparison with Newton method using a line search strategy (standard Armijo rule). The trial step is of the form $s_k = \alpha_k s_k^Q$ where s_k^Q is the solution of the the the subproblem (9), and the step length $\alpha_k > 0$ is chosen such as

$$f(x_k + s_k) \leq f(x_k) + 10^{-3} g_k^T s_k,$$

The appropriate value of α_k is estimated using a backtracking approach with a contraction factor set to 0.9 and where the step length is initially chosen to be 1. The Newton method with the line search will be called `NEWTON-LS`.

The solvers `TR-EN`/`ARC-EN` and `NEWTON-LS` are compared with TR/ARC using the Lanczos-based solver `GLTR`/`GLRT` implemented in `GALAHAD` [12]. The two subproblem solvers are coded in Fortran and interfaced with Matlab using the default parameters. The matrix M_k in the scaled norms is fixed to identity matrix. The other parameters defining the original `TR-GLTR`/`ARC-GLRT` method (Algorithm 1) and the proposed `TR-EN`/`ARC-EN` algorithm (Algorithm 2) are chosen as described in [6]. We add also in our TR comparison the `TR-DOGLEG` method of Powell [18] as its computational cost for solving the subproblem is comparable with that of `TR-EN`.

For all algorithms the maximum number of "outer iterations", that are iterations in the unconstrained minimizer, is set to 100000. Similarly to `NEWTON-LS`, only successful outer iterations of `TR-EN`/`ARC-EN` algorithm will be counted as outer iterations. In fact, unsuccessful iterations in Algorithm 2 can be seen as a specific backtracking strategy to satisfy the acceptance criterion $\rho_k \geq \eta_1$. All the algorithms stop when

$$\|g_k\| \leq \epsilon \qquad \text{with} \quad \epsilon = 10^{-5}.$$

We use the Moré/Garbow/Hillstrom test problems [15] which are implemented in Matlab. All the test problems are smooth and have a least-squares structure (meaning that the objective function is of the form $f(x) = \frac{1}{2}\|F(x)\|^2$ where $F : \mathbb{R}^n \mapsto \mathbb{R}^m$). Unlike the CUTEr test problems, the Jacobian matrix of the residual function F for all the test problems [15] is available in Matlab. Thus, one can obtain locally convex models (i.e., symmetric positive definite approximation of the local Hessian) using a Gauss-Newton approximation of the type:

$$B(x) = J_F(x)^T J_F(x) + \epsilon_B I_n,$$

where $J_F(x)$ is the Jacobian matrix of the residual vector function F at the point x, I_n is the identity matrix. ϵ_B is a regularization parameter used to avoid a possible singularity of the matrix B, in our experiments we fixed $\epsilon_B = 10^{-5}$.

The problems description is reported in Table 1. We use the standard starting points suggested in [15] for all the problems. To have a large bed-test, we create additional problems by varying the problem dimension n when it is possible (typically $n = 10, 50, 100, 200$ and 300).

Dimension n of problems	$2 \leq n \leq 9$	$10 \leq n \leq 20$	$21 \leq n \leq 50$	$51 \leq n \leq 300$
# of problems	22	13	8	19

Table 1: The distribution of the number of variables of the problems in the test set.

To compare the performance of the algorithms we use performance profiles proposed by Dolan and Moré [10] 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 $\rho(\tau)$ of a solver s is defined as the fraction of problems where the performance ratio $r_{p,s}$ is at most τ

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

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

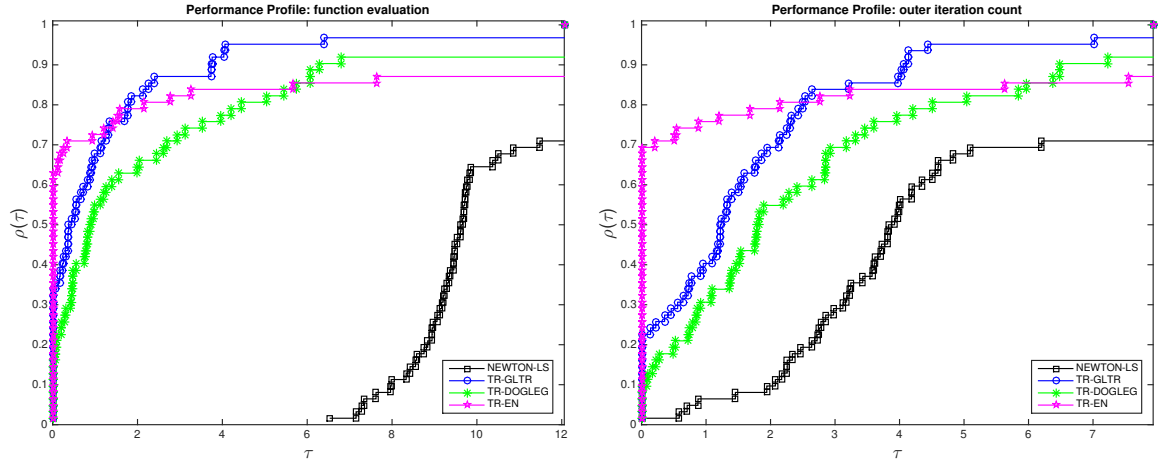
$$r_{p,s} \ = \ \frac{t_{p,s}}{\min\{t_{p,s} : s \in \mathcal{S}\}},$$

where $t_{p,s} > 0$ measures the performance of the solver s when solving problem p (seen here as the function evaluation and the outer iteration number). Better performance of the solver s, relatively to the other solvers on the set of problems, is indicated by higher values of $\rho(\tau)$. In particular, efficiency is measured by $\rho(1)$ (the fraction of problems for which solver s performs the best) and robustness is measured by $\rho(\tau)$ for τ sufficiently large (the fraction of problems solved by s). Following what is suggested in [10] for a better visualization, we will plot the performance profiles in a $\log_2$-scale (for which $\tau = 1$ will correspond to $\tau = 0$).

5.2 Results

We consider first the TR algorithms. In Figure 1 the performance profiles show that `TR-EN` improves the efficiency of the TR algorithms on the tested problems. In fact, the outer iteration performance profiles (see Figure 1(b)) show that the use of the energy norm leads to a significant improvement on terms of the efficiency (for $\tau = 0$, on 70% of the tested problems `TR-EN` performs the best, 23% for `TR-GLTR`, and around 10% for `TR-DOGLEG`). The use of the l_2 norm leads to a better robustness (i.e., `TR-GLTR` and `TR-DOGLEG` are solving a large percentage of the tested problem for large τ values). The same analysis applies in terms of function evaluation (see Figure 1(a)) where the efficiency of `TR-GLTR` and `TR-DOGLEG` is slightly improved compared to the out iterations performance profiles (but not as good as `TR-EN`). `NEWTON-LS` method is performing the worst on the tested problems, both in terms of efficiency and robustness.

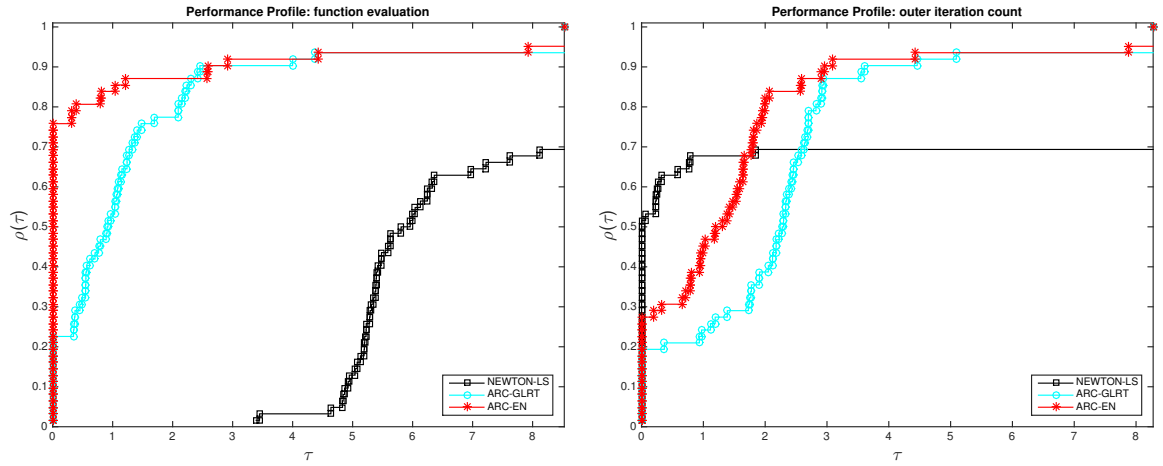
For the ARC algorithms, see Figure 2, the performance profiles show that the use of the energy norm (i.e., `ARC-EN`) improves the efficiency of the ARC algorithms over the tested problems. Unlike the TR performance profiles, `NEWTON-LS` is the most efficient method in terms of the



(a) Function evaluation. (b) Outer iteration.

Figure 1: TR performance profiles for the 62 optimization problems under consideration.

outer iteration number, see Figure 2(b). In fact, on over 50% of the tested problems `NEWTON-LS` performs the best, while `ARC-EN` solves 28% and `ARC-GLTR` around 20%. Regarding the function evaluation performance profile, see Figure 2(a), `NEWTON-LS` method looses its efficiency due to the backtracking strategy employed for each outer iteration. `ARC-EN` outperforms (by far) both solvers on terms of efficiency (in over 76% of the tested problems `ARC-EN` performs the best, and around 24% for `ARC-GLTR`). In terms of robustness, `ARC-GLTR` and `ARC-EN` exhibit similar performance. Again, as for TR algorithms, `NEWTON-LS` is performing the worst on the tested problems regarding the robustness.



(a) Function evaluation. (b) Outer iteration.

Figure 2: ARC performance profiles for the 62 optimization problems under consideration.

5.3 Test problems with many degrees of freedom

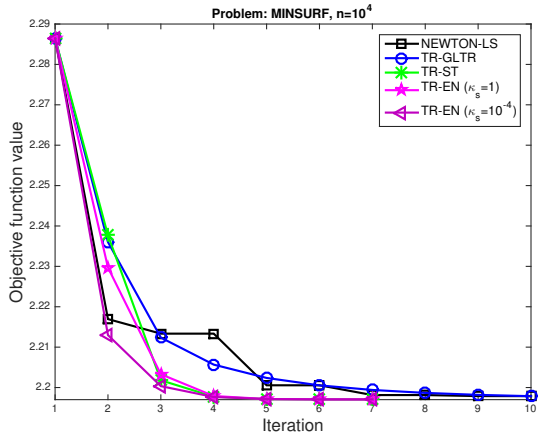
In this section we solve three large scale convex optimization problems from [13]. The first one is called MINSURF and consists of finding the surface with minimal area that lies above an obstacle with given boundary conditions. The second optimization problem, called IP, is an inverse problem from image processing. The third test case, called IGNISC, is an optimal-control problem related to a solide-ignition model. MINSURF and IGNISC are tested with $n = 10000$, for the IP optimization our tests are performed using $n = 3969$. The initial iterate for all the tested algorithms and problems is set to zero.

In our previous numerical experiments we solved exactly the subproblem (9). However, for large scale optimization problems finding the exact solution may be out of reach. In this section for TR-EN, ARC-EN and NEWTON-LS, such case is geared by mean of the truncated CG method. The solvers TR-EN/ARC-EN and NEWTON-LS will be compared to TR-GLTR/ARC-GLRT. For the TR comparison, the TR-dogleg approach will be replaced by TR-ST which denotes a TR method that uses the Steihaug-Toint approach [7] to approximate the solution of TR subproblems. The TR-ST is the most commonly used in TR methods for approximating the solution of large scale TR sub-problems. The subproblem solvers GLTR and GLRT are tested using the default parameters (as in GALAHAD [12]). The NEWTON-LS and TR-ST subproblems are declared to be solved when the relative residual error is getting less than 10^{-4}.

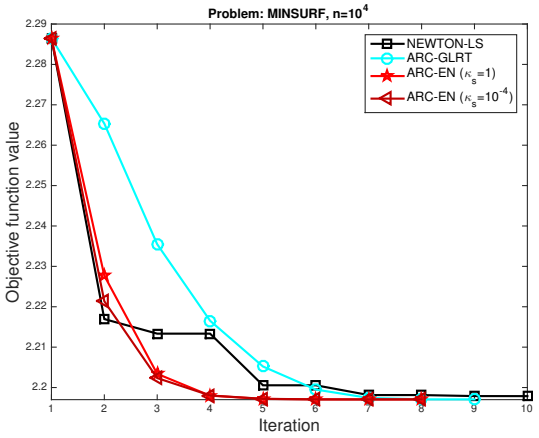
For TR-EN/ARC-EN, we use a stopping criterion of the form (26). To quantify the sensibility of such termination condition (i.e., s-rule) two different values of κ_s (1 and 10^{-4} are tested). Each iteration of the steps 1, 2 and 3 of TR-EN/ARC-EN (see Algorithm 2) formed an outer iteration. Within each outer iteration, we must solve the minimization problem involving $m_k^Q(s)$ at step 2 by means of the aforementioned truncated CG method. This procedure will be called the inner iteration minimization.

Figure 3 depicts the values of the objective function over outer iterations during the application of TR/ARC algorithms on the three large scale optimization problems. For all the tested algorithms the variation of the objective function value is more significant at the early stages of the optimization process. The TR-EN/ARC-EN algorithms outperform the other methods on the tested problems. In fact, for the three optimization problems, the use of the energy norm ensures a better decrease on the objective function value with a faster convergence speed. For the TR comparison, TR-ST and TR-EN (with $\kappa_s = 10^{-4}$) give comparable performance in terms of outer iterations and converge faster than the other methods on the tested problems (i.e., in less than 10 outer iterations).

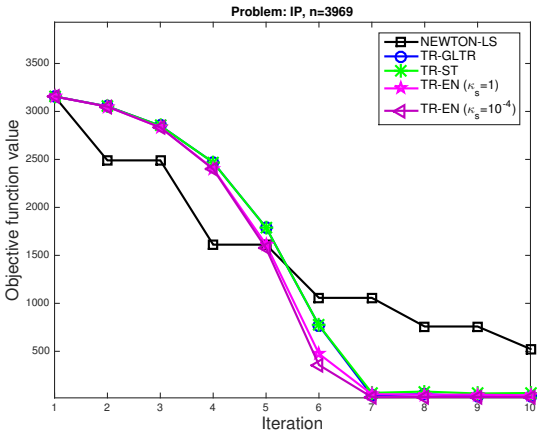
The TR-EN/ARC-EN algorithms are performing better using an accurate termination condition, i.e., $\kappa_s = 10^{-4}$. A high accuracy on the termination conditions can be computationally very demanding in terms of the inner iterations number. Table 2 outlines such a statement on the tested problems, it shows the variation of the number of inner iterations during the application of TR-EN/ARC-EN algorithms. As expected, for a large value of κ_s, the number of inner iterations is by far less than the number required when using a tight termination condition. We note also that, for the tested problems, the choice of κ_s is problem dependent. In fact, for MINSURF and IP problems the obtained results with the two different accuracies $\kappa_s = 1$ and 10^{-4} are still very close, meaning that the choice of κ_s does not affect significantly the number of outer iterations. However, for IGNISC problem, the accuracy of the termination condition affects the speed of convergence.



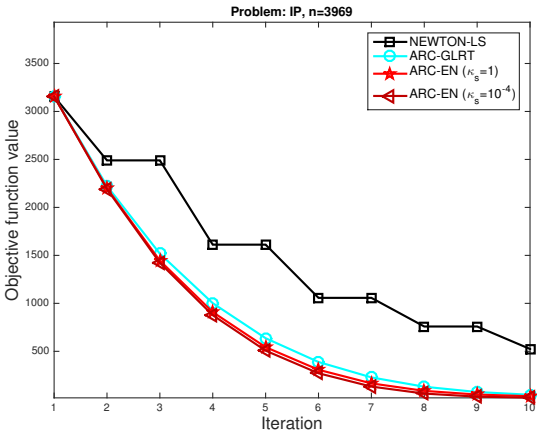
(a) TR algorithms.



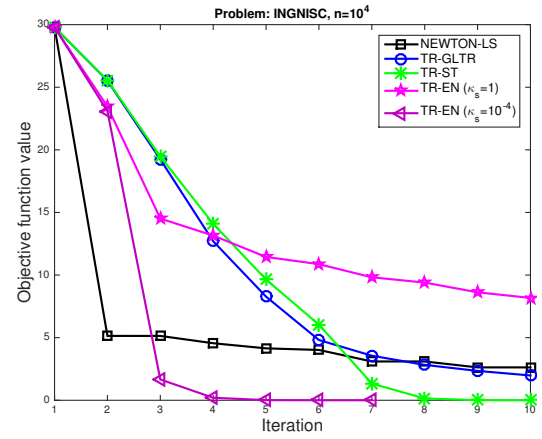
(b) ARC algorithms.



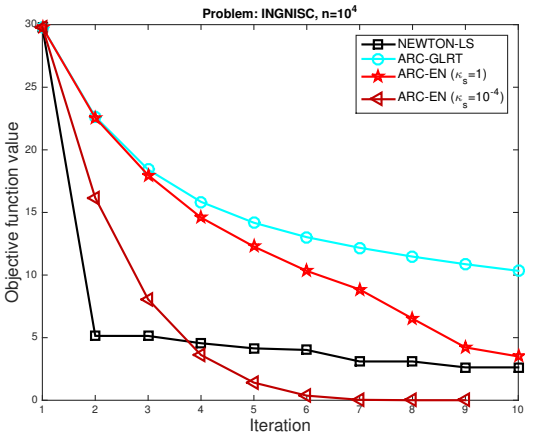
(c) TR algorithms.



(d) ARC algorithms.



(e) TR algorithms.



(f) ARC algorithms.

Figure 3: Objective function values during the application of TR/ARC algorithms on large scale optimization problems.

Table 2: Objective function values and the number of inner iterations during the application of TR-EN/ARC-EN algorithm on large scale optimization problems.

	Outer iter.	TR-EN				ARC-EN			
		$\kappa_s = 1$		$\kappa_s = 10^{-4}$		$\kappa_s = 1$		$\kappa_s = 10^{-4}$	
		$f(x_k)$	# Inner iter.	$f(x_k)$	# Inner iter.	$f(x_k)$	# Inner iter.	$f(x_k)$	# Inner iter.
MINSURF	1	2.2865	11	2.2865	166	2.2865	16	2.2865	172
	2	2.2297	17	2.2130	162	2.2277	19	2.2214	188
	3	2.2033	31	2.2003	229	2.2034	30	2.2023	227
	4	2.1979	50	2.1977	247	2.1980	48	2.1980	239
	5	2.1972	70	2.1971	266	2.1972	67	2.1972	260
	6	2.1971	126	2.1971	283	2.1971	85	2.1971	280
IP	1	3050.32	6	3050.32	22	2199.5	1	2190.0	14
	2	2833.97	3	283.388	19	1445.0	1	1421.8	60
	3	2404.14	1	239.794	32	908.64	1	876.21	98
	4	1614.05	1	15.7374	64	543.42	1	506.95	92
	5	474.356	1	35.7950	51	307.75	1	271.33	99
	6	52.9156	1	22.2981	40	165.85	1	132.55	108
IGNISC	1	29.790	18	23.085	141	22.560	1	16.164	121
	2	23.456	1	1.6612	95	17.979	1	8.0594	138
	3	14.514	1	0.2009	161	14.592	2	3.6155	161
	4	13.152	2	0.0106	219	12.274	2	1.4081	189
	5	11.446	1	0.0030	317	10.329	3	0.3685	208
	6	10.869	2	0.0029	443	8.8495	3	0.3702	246

6 Conclusions and perspectives

In this paper we have investigated the use of the energy norm in the TR/ARC framework, when the Hessian of the model is positive definite (Assumption 2.1). With this norm choice, a significant reduction on the computational cost of the related subproblem is theoretically explained and numerically showed. In fact, for successful iterations occurring in the minimization, finding the subproblem solution costs the same as solving a linear system. The unsuccessful outer iterations are getting comparatively inexpensive, as they just require an additional energy norm evaluation. For large scale problems, we propose to use a truncated-CG approximation to compute an approximate solution for the subproblem.

The proposed approaches are in practice line search algorithms with a special backtracking strategy and an acceptability condition in the spirit of TR/ARC methods. Moreover, we showed that the proposed line search algorithm, when it is derived by the ARC algorithm, enjoys a worst-case iteration complexity of $\mathcal{O}(\epsilon^{-3/2})$. Preliminary numerical experiments with basic implementations showed encouraging performance. An additional feature of the algorithm is that it enables also the use of a preconditioner which can be necessary when dealing with an ill-conditioned model Hessian.

For `TR-EN/ARC-EN` algorithms (Algorithm 2), we assumed that a positive definite local approximation of the model Hessian is available (Assumption 2.1). Such assumption is required

to define the energy norm. The generalization of `TR-EN/ARC-EN` algorithms for all local approximations of the Hessian as well as an extensive numerical experiments will be addressed in a forthcoming work.

Our main motivation is to use `TR-EN/ARC-EN` algorithms to solve data assimilation problems (e.g., ocean and meteorology data assimilation systems [8, 19]). Such problems are typically extremely large and very costly but with a positive definite local approximation of the Hessian. Applying the proposed approaches can be very convenient to define an efficient solver for this kind of problems and will be the topic of a future work.

References

[1] M. Arioli, D. Loghin, and A. J. Wathen. Stopping criteria for iterations in finite element methods. *Numer. Math.*, 99(3):381–410, 2005.

[2] M. Arioli, E. Noulard, and A. Russo. Stopping criteria for iterative solvers. *Calcolo*, 38:97–112, 2001.

[3] S. Bellavia, B. Morini, C. Cartis, N. I. M. Gould, and Ph. L. Toint. Convergence of a regularized euclidean residual algorithm for nonlinear least-squares. *SIAM J. Numer. Anal.*, 48(1):1–29, 2010.

[4] T. Bianconcini, G. Liuzzi, B. Morini, and M. Sciandrone. On the use of iterative methods in cubic regularization for unconstrained optimization. *Comput. Optim. Appl.*, 60(1):35–57, 2014.

[5] 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.

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

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

[8] P. Courtier, J.-N. Thépaut, and A. Hollingsworth. A strategy for operational implementation of 4D-Var, using an incremental approach. *Q. J. R. Meteorol. Soc.*, 120:1367–1388, 1994.

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

[11] N. I. M. Gould, S. Lucidi, M. Roma, and Ph. L. Toint. Solving the trust-region subproblem using the Lanczos method. *SIAM J. Optim.*, 9(2):504–525, 1999.

[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.*, 29:353–372, 2003.

[13] S. Gratton, M. Mouffe, A. Sartenaer, Ph. L. Toint, and D. Tomanos. Numerical experience with a recursive trust-region method for multilevel nonlinear bound-constrained optimization. *Optim. Methods Softw.*, 25(3):359–386, 2010.

[14] 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.

[15] J. J. Moré, B. S. Garbow, and K. E. Hillstrom. Testing unconstrained optimization software. *ACM Trans. Math. Softw.*, 7:17–41, 1981.

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

[17] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006.

[18] M. J. D. Powell. A new algorithm for unconstrained optimization. *In: J. B. Rosen, O. L. Mangasarian and K. Ritter (eds) Nonlin. Program.*, pages 31–65, 1970.

[19] Y. Trémolet. Model error estimation in 4D-Var. *Q. J. R. Meteorol. Soc.*, 133:1267–1280, 2007.