To appear in *Optimization Methods & Software*
Vol. 00, No. 00, Month 20XX, 1–14

ARC$_q$: a new Adaptive Regularization by Cubics variant

Jean-Pierre Dussault

(*Received 00 Month 20XX; final version received 00 Month 20XX*)

We present a new Adaptive Regularization by Cubics (ARC) algorithm, which we name ARC$_q$ and which is very close in spirit to trust region methods, closer than the original ARC is. We prove global convergence to second-order critical points. We also obtain as a corollary the convergence of the original ARC method. We prove the optimal complexity property for the ARC$_q$ and identify the key elements which allow it. We end by proposing an efficient implementation using a Cholesky like factorization. Limited preliminary experimentation suggests that ARC$_q$ may be more robust than its trust region counterpart and that our implementation is efficient.
Keywords: Nonlinear optimization — unconstrained optimization — Trust region algorithms – Adaptive cubic regularization methods

Introduction

We consider the problem

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

where $\boxed{f : \mathcal{C}^2(\mathbb{R}^n \to \mathbb{R})}$. No convexity assumption is made and we address the problem of computing a local minimum of f. Actually, we consider algorithms that compute a point x^* satisfying (approximately) the following necessary optimality conditions: x^* is

- a stationary point of f ($\nabla f(x^*) = 0$), first order critical point and
- with a positive semi-definite hessian matrix ($\nabla^2 f(x^*) \succeq 0$), second order critical point.

While designing an algorithm to ensure its convergence to such second-order solutions is not straightforward, two main families of algorithm have been developed, namely the *line search* (LS) and the *trust region* (TR). The trust region family provides a simpler algorithmic framework to achieve both first ($\nabla f(x^*) = 0$) and second ($\nabla^2 f(x^*) \succeq 0$) order optimality. Line search algorithms achieve first order optimality ($\nabla f(x^*) = 0$) and may be extended to ensure second-order optimality, but resulting in an arguably more complex method. Both families are actually descent algorithms, the descent of the objective function at each iteration allowing to prove convergence to a critical point from any starting point x_0.

Efficiency of such iterative descent algorithms is often analyzed from the point of view of asymptotic convergence speed. Recent results used the *worst case complexity* measure to analyze the efficiency of descent algorithms. This measure builds on the global convergence property. Since the sequence produced by a globally convergent algorithm possess cluster points satisfying the necessary optimality conditions, it is natural to bound the maximum number of iterations before some approximate necessary condition is satisfied.

Recently, following the trust region paradigm, cubic regularization algorithms [5, 6, 19] among which the ARC were introduced and shown to improve its worst case complexity. ARC possesses the best worst case complexity to solve (1).

In this paper, we introduce ARC$_q$, a new ARC variant. Since it is closer to trust region algorithms than the original ARC, we present an unified framework, somewhat in the spirit of [22] and [12], where ARC is analyzed in a TR framework. Those variants use a model of f to both compute a step and decide if the resulting step is successful. In contrast, we use the model only to compute the step and revert to a quadratic Taylor approximation of f to assess its success, as it is done in trust region methods.

We will provide simple proofs to ensure that at least one limit point of the generated sequence satisfy the first and second order necessary optimality conditions. Our analysis follows the lines of Fletcher's analysis [10]. As we will see, the proof actually allows to claim the stronger result that any cluster point is second-order stationary for ARC$_q$ and, as a corollary, ARC. The originality of our analysis does not lie in weakening assumptions to obtain the convergence results but in a very simple analysis. The overall analysis of ARC$_q$, its global convergence to second order critical points and its optimal complexity to compute first order critical points is remarkably concise.

We adopt the convention that x, d are column vectors of $\mathbb{R}^n$ and $\nabla f(x)$ is a row vector. The algorithms use a second order Taylor approximation to f

$$q_x(d) = f(x) + \nabla f(x)d + \frac{1}{2}d^t\nabla^2 f(x)d.$$

A regularized model is minimized to obtain a direction d. The trust region approach minimizes the model in a *trust region* $D := \{d : \|d\| \leq \delta\}$:

$$\min_{d\in D} q_x(d) \tag{2}$$

which is equivalent to write

$$d \in \arg\min T_x^\delta \overset{\text{def}}{=} q_x(d) + \chi_\delta(d)$$

where $\chi_\delta(d)$ is the characteristic function of the convex set $D \subseteq \mathbb{R}^n$:

$$\chi_\delta(d) = \begin{cases} 0 & \|d\| \leq \delta \\ +\infty & \|d\| > \delta \end{cases}.$$

ARC minimizes a cubic model [14]

$$\min c_x^\alpha \overset{\text{def}}{=} q_x(d) + \frac{\sigma}{3}\|d\|^3 = q_x(d) + \frac{1}{3\alpha}\|d\|^3. \tag{3}$$

The parameter $\alpha = \frac{1}{\sigma} \in \mathbb{R}^+$ in ARC plays a similar role as the parameter $\delta \in \mathbb{R}^+$ in the trust region method.

Although it is somewhat artificial to view the trust region as a regularization, it is indeed the limit when the polynomial order goes to infinity of such higher order regularizations addressed in [3].

When discussing aspects common to ARC and TR, we will use β to denote either δ or α and $m_x^\beta(d)$ either the $T_x^{\beta=\delta}$ trust region model or $c_x^{\beta=\alpha=\frac{1}{\sigma}}$ the adaptive cubic model.

The analogy between TR and ARC is further explored when analyzing the solutions (global minimizers) of the subproblems (2) and (3). Both possess a global minimizer d_β satisfying the condition

$$\nabla f(x) + d_\beta^t(\nabla^2 f(x) + \lambda I) = 0 \tag{4}$$

for a suitable value $\lambda \geq \lambda_{min} \geq 0$, i.e. both are solution to a Levenberg-Marquardt like system of linear equations. $\lambda \geq \lambda_{min}$ ensures that $(\nabla^2 f(x) + \lambda I) \succeq 0$. Therefore, both TR and ARC compute among the trajectory of solutions $d_\beta(\lambda)$ of (4) a suitable point $d(\bar{\lambda})$ for some $\bar{\lambda} \geq \lambda_{min}$.

TR chooses $\bar{\lambda}$ such that $\bar{\lambda}(\delta - \|d_\delta(\bar{\lambda})\|) = 0$, i.e. $d_\delta(\bar{\lambda})$ is either the Newton's direction $\nabla^2 f(x)^{-1} \nabla f(x)^t$ if it lies in the trust region and $\bar{\lambda} = 0$, or a point lying on the boundary of the region with $\bar{\lambda} \geq \lambda_{min}$.

ARC chooses $\bar{\lambda} = \frac{\|d_\alpha(\bar{\lambda})\|}{\alpha}$ which never corresponds to the Newton's direction but may approach it when d is small and α is not, which results in a small value of $\bar{\lambda}$.

From the discussion above, we see that TR and ARC are similar, but do not produce the same iterates, TR usually reducing to Newton's method close to a non degenerate minimizer of f. When subproblem (3) is solved to an approximate global minimum, the worst case complexity of ARC is $\mathcal{O}(\epsilon^{-\frac{3}{2}})$, the best bound known for solving (1).

The unified algorithm we study in this paper is essentially the trust region method. In each iteration, the search direction is chosen to be the solution of either the trust region model $T_x^\beta(d)$ or the adaptive cubic model $c_x^\beta(d)$. In contrast to [5, 6] or the unified framework in [22] and [12], the classification of the search direction as a successful, very successful and unsuccessful step is now based on the Taylor quadratic approximation q, which is why we abbreviate this method by ARC$_q$.

```
Model_Algorithm($x, \beta, f, m$)
  { Given: (initial) $x$, $\beta > 0$; }
  {         parameters $0 < \gamma_1 < \gamma_2 \leq 1 < \gamma_3$; }
  {         parameters $0 < \eta_1 < \eta_2 < 1$; }
  {         objective function $f$ and model $m$. }
  while ( $\neg$ termination_criterion )
    $d \leftarrow$ Solve_Model($m_x^\beta$)
    $\Delta f \leftarrow f(x) - f(x+d)$
    $\Delta q \leftarrow q(0) - q(d)$
    $r \leftarrow \frac{\Delta f}{\Delta q}$
    if ($r < \eta_1$ ) then $\beta \leftarrow \gamma_1 \beta$        {Unsuccessful}
    else
      $x \leftarrow x + d$
      if ($r < \eta_2$ ) then $\beta \leftarrow \gamma_2 \beta$   {Successful}
      else $\beta = \gamma_3 \beta$                       {Very successful}
  Result $\leftarrow x$
```

Algorithm 1: Abstract algorithm.

As noted above, solving the model involves solving a Levenberg-Marquardt like linear system for a suitable value of the λ parameter. One classical approach is to use some kind of iterative algorithm to compute the value for the parameter λ. In this approach, a scalar function having a root at the value $\bar{\lambda}$ is used together with a root finding algorithm. Derivatives of this function usually involve a factorization of $\nabla^2 f(x)$, making the approach limited to rather small scaled problems. We adapt a technique from [11] to propose an implementation potentially suitable for really large problems when the hessian matrix is sparse and its LDL^t factorization may be obtained cheaply.

The paper is organized as follows. We state in section 1 necessary optimality conditions on which our global convergence analysis relies. We then give in section 2 two preliminary lemmas useful both in the convergence and complexity analysis. In section 3, we address the global convergence of the unified scheme. We provide an analysis unifying the TR and ARC and ARC$_q$ frameworks. However, the result for ARC$_q$ is stronger and obtained using a very simple analysis. We address the worst case complexity issues in section 4 and provide a unified implementation framework together with some preliminary numerical observations in section 5.

1. Necessary optimality conditions

We first show that a necessary condition for x^* to be a local minimum is that $d = 0$ is a minimizer of the models $m^{\beta}_{x^*}(d)$. Actually, we show that the usual necessary conditions $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succeq 0$ are equivalent to the fact that $d = 0$ is a minimizer of the model at x^*.

Theorem 1.1 (Second order necessary conditions) *If x^* is a local minimizer of f, then the following three statements are equivalent necessary optimality conditions.*

(1) $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succeq 0$;
(2) $\forall \delta > 0$, $d = 0$ is a global minimizer of q_{x^*} subject to $\|d\| \le \delta$;
(3) $\forall \alpha > 0$, $d = 0$ is an unconstrained global minimizer of $c^{\alpha}_{x^*}$.

Proof. $1 \Rightarrow 2$ We write

$$q_{x^*}(d) - f(x^*) = \nabla f(x^*)d + \frac{1}{2} d^t \nabla^2 f(x^*) d \ge 0, \ \forall d \in \mathbb{R}^n$$

so that $d = 0$ reaches this lower bound.

$1 \Rightarrow 3$

$$c^{\alpha}_{x^*}(d) - f(x^*) = q_{x^*}(d) - f(x^*) + \frac{1}{3\alpha}\|d\|^3 \ge 0, \ \forall d \in \mathbb{R}^n,$$

and here again $d = 0$ reaches the lower bound.

2 or 3 $\Rightarrow$ 1 At $d = 0$, the gradient and Hessian matrix of the models are precisely $\nabla f(x^*)$ and $\nabla^2 f(x^*)$.

■

2. Preliminary results

For both variants, a global minimizer d_k of the model $m^{\beta_k}_{x_k}(d)$ satisfies [5, Theorem 3.1], [20, Theorem 4.3]

$$\begin{aligned} \nabla f(x_k) + d_k^t \left(\nabla^2 f(x_k) + \lambda_k I\right) &= 0, \qquad (5) \\ \nabla^2 f(x_k) + \lambda_k I &\succeq 0. \qquad (6) \end{aligned}$$

For the ARC using $\beta_k = \alpha_k$, from [5, Theorem 3.1], $\lambda_k = \frac{\|d_k\|}{\alpha_k}$. For the TR, whenever $\lambda_k > 0$, $\|d_k\| = \beta_k$, i.e. the radius of the trust region is active when its KKT multiplier λ_k does not vanish. An important difference between ARC's and the trust-region directions is apparent by observing that the equation (5), for fixed α, biases the resulting direction more and more on the steepest descent as $\|d\|$ is larger.

This characterization of d_k yields the following.

Lemma 2.1 *(1) [20, Theorem 4.6] Assume d_k is a minimizer of the trust region model $q_{x_k}(d)$. Then,*

$$\Delta q_{x_k}(d_k) = f(x_k) - q_{x_k}(d_k) \ge \frac{1}{2}\|\nabla f(x_k)\| \min\left(\delta_k, \frac{\|\nabla f(x_k)\|}{\|\nabla^2 f(x_k)\|}\right).$$

(2) Assume d_k is a minimizer of the cubic model $c^{\alpha_k}_{x_k}(d)$. Then,

$$\Delta q_{x_k}(d_k) = f(x_k) - q_{x_k}(d_k) \geq \frac{\|d_k\|^3}{2\alpha_k},$$

and

$$\Delta m^{\alpha_k}_{x_k}(d_k) = f(x_k) - m^{\alpha_k}_{x_k}(d_k) \geq \frac{\|d_k\|^3}{6\alpha_k}.$$

Proof. of 2. $f(x_k) - q_{x_k}(d_k) = -(\nabla f(x_k)d_k + \frac{1}{2}d_k^t\nabla^2 f(x_k)d_k)$ and using (5), $-(\nabla f(x_k)d_k + d_k^t\nabla^2 f(x_k)d_k) = \lambda_k\|d_k\|^2$, which using (6) combines to $f(x_k) - q_{x_k}(d_k) \geq \frac{\|d_k\|^3}{2\alpha_k}$. ∎

We next observe that whenever $\nabla^2 f$ is Lipschitz continuous (with constant L_H), α_k is actually bounded away from zero for the ARC$_q$ variant.

Lemma 2.2 *If $\nabla^2 f$ is Lipschitz continuous (with constant L_H), then when $\alpha_k < \frac{1-\eta_2}{L_H}$, $\alpha_{k+1} \geq \alpha_k$. Thus, $\alpha_k \geq \min(\alpha_0, \gamma_1\frac{1-\eta_2}{L_H}) \overset{\text{def}}{=} \frac{1}{L_0}$ for all $k \geq 0$.*

Proof. We write $r_k = \frac{\Delta f_{x_k}(d_k)}{\Delta q_{x_k}(d_k)}$ as

$$r_k = 1 + \frac{q_{x_k}(d_k) - f(x_k + d_k)}{\Delta q_{x_k}(d_k)}$$

and noting from Lemma 2.1 that $\Delta q_{x_k}(d_k) \geq \frac{\|d_k\|^3}{2\alpha_k}$ while $|q_{x_k}(d_k) - f(x_k + d_k)| \leq L_H\|d_k\|^3$, we get that $r_k > \eta_2$ whenever $\alpha_k < \frac{1-\eta_2}{L_H}$ and thus $\alpha_k \geq \gamma_1\frac{1-\eta_2}{L_H}$. ∎

For trust regions, a similar result (see for example [8, Theorem 6.4.2 and 6.4.3]) holds only locally in some neighborhood of a non stationary point such that $\|\nabla f(\bar{x})\| \geq \epsilon > 0$. The result here is much stronger, allowing better convergence results as we now develop.

3. Global convergence

We will now analyze the cluster points of the TR and ARC$_q$ algorithms and show that at least one cluster point satisfies the necessary conditions of Theorem 1.1. We state first one common theorem which uses a parameter β corresponding to α for the ARC$_q$ and to δ for the TR. The sequence generated by the algorithm is denoted by $\{x_k\}$ and the appropriate algorithmic quantities $\beta_k = (\delta_k \text{ or } \alpha_k)$, d_k, $\Delta q_k = \Delta q_{x_k}(d_k) = q_{x_k}(0) - q_{x_k}(d_k)$, $\Delta f_k = \Delta f(x_k) = f(x_k) - f(x_k + d_k)$, $r_k = r(x_k) = \frac{\Delta f_k}{\Delta q_k}$ are labeled accordingly. Similarly, $m^{\beta}_x(d)$ corresponds to either $q_x(d)$ or $c^{\alpha}_x(d)$.

Theorem 3.1 *Let $\{x_k\}$ be generated by algorithm 1. We assume that f is $\mathcal{C}^2(\mathbb{R}^n)$, that $f(x_k)$ is bounded below and that the generated sequence $\{x_k\}$ remains in a compact $C \subset \mathbb{R}^n$; then there exists a cluster point of the generated sequence $\{x_k\}$ which satisfies the second order necessary optimality conditions.*

Proof. The compacity assumption ensures that any subsequence possesses a cluster point. Following [10, Theorem 5.1.1, page 96], we argue that we may reduce the analysis to one of two cases: either the sequence $\{\beta_k\}$ remains bounded away from zero, in which case we may extract a subsequence such that

(1) $\beta_{k_i} \geq \bar{\beta} > 0$ and $r_{k_i} \geq \eta_1 \; \forall i$,

or some subsequence of $\{\beta_k\}$ converges to zero and we may extract a subsequence such that

(2) $r_{k_i} < \eta_1$, $\forall i$ and $\beta_{k_{i+1}} \to 0$.

(1) Let us show that any cluster point $\bar{x}$ of such a subsequence satisfies the necessary conditions of Theorem 1.1. Successful or very successful steps strictly decrease $f(x)$ while unsuccessful ones keep it constant. Therefore, $f(x_k)$ is bounded below and monotonically decreasing, it is a Cauchy sequence and converges to some finite value and thus $\Delta f_k \to 0$ which implies $\Delta q_{k_i} \to 0$ since $r_{k_i} \geq \eta_1$ is bounded away from zero.

Now consider $\bar{d}$ an optimal solution of $\min m_{\bar{x}}^{\bar{\beta}}(d)$, $\hat{x} = \bar{x} + \bar{d}$ and $\hat{d}_{k_i} = \hat{x} - x_{k_i}$. Since d_k minimizes $m_{x_k}^{\beta_k}(d)$, we get

$$m_{x_{k_i}}^{\bar{\beta}}(\hat{d}_{k_i}) \geq m_{x_{k_i}}^{\beta_{k_i}}(\hat{d}_{k_i}) \geq m_{x_{x_i}}^{\beta_{k_i}}(d_{k_i}) = f(x_{k_i}) - \Delta m_{x_{k_i}}^{\beta_{k_i}}. \tag{7}$$

We will see that $\Delta m_{x_{k_i}}^{\beta_{k_i}} \to 0$, which allows to write, taking the limit in (7)

$$m_{\bar{x}}^{\bar{\beta}}(\bar{d}) \geq f(\bar{x}) = m_{\bar{x}}^{\bar{\beta}}(0),$$

which justifies the claim that $\bar{x}$ satisfies the optimality conditions of Theorem 1.1.

Let us now prove that $\Delta m_{x_{k_i}}^{\beta_{k_i}} \to 0$. For the trust region variant, we have $\Delta m_{x_{k_i}}^{\beta_{k_i}} = \Delta q_{x_{k_i}} \to 0$ directly. For the ARC$_q$variant, observing that

$$0 \geq -\Delta m_{x_{k_i}}^{\beta_{k_i}} = -\Delta q_{k_i} + \frac{1}{3\alpha_{k_i}} \|d_{k_i}\|^3 \geq -\Delta q_{k_i},$$

we deduce $-\Delta m_{x_{k_i}}^{\beta_{k_i}} \geq -\Delta q_{k_i}$, taking the limit, recalling that $\Delta q_{k_i} \to 0$, we get $\Delta m_{x_{k_i}}^{\beta_{k_i}} \to 0$.

(2) Since f is $\mathcal{C}^2(\mathbb{R}^n)$ and the iterates assumed to lie in some compact set, $\nabla^2 f$ satisfies a Lipschitz condition (with unknown L_H); therefore, for the ARC$_q$ variant, this case is impossible, contradicting the Lemma 2.2.

For the TR variant, the analysis in [10, Theorem 5.1.1, page 96] shows that this case cannot occur close to $\bar{x}$ which is not stationary ($\nabla f(\bar{x}) \neq 0$) or not optimal ($\nabla^2 f(\bar{x}) \not\succeq 0$).

■

Now, since the ARC$_q$ variant may not produce the second case, we are able to improve upon the result above.

Corollary 3.2 *Let $\{x_k\}$ be generated by algorithm 1 for the model c_x^{α}. We assume that f is $\mathcal{C}^2(\mathbb{R}^n)$ and $\nabla^2 f$ satisfies a Lipschitz condition, that $f(x_k)$ is bounded below; then any cluster point of the generated sequence $\{x_k\}$ satisfies the second order necessary optimality conditions.*

Proof. Since $\alpha_k \geq \bar{\alpha} > 0$, whenever the sequence possess a cluster point, we always have a subsequence of successful iterates satisfying the condition (1) in the Theorem 3.1. ■

The above analysis suggests how to obtain the global convergence property of the original ARC in a simple way.

Corollary 3.3 *Let $\{x_k\}$ be generated by the ARC algorithm, that is Δq is replaced by $\Delta c_x^{\alpha} = c_x^{\alpha}(d) - c_x^{\alpha}(0)$ in algorithm 1. If $f(x_k)$ is bounded below and $\nabla^2 f$ satisfies a lipschitz* *condition; then any cluster point of the generated sequence $\{x_k\}$ satisfies the second order necessary optimality conditions.*

Proof. In the proof of part (1) of the Theorem 3.1, replacing Δq_{k_i} by $\Delta c_{x_{k_i}}^{\alpha_{k_i}}$ actually allows to get directly the desired conclusion. ■

4. Complexity bounds

We now address the worst-case complexity issues of ARC$_q$.

Since ARC$_q$ uses the same directions as ARC, only the analysis of [6] that concerns the selection of successful, unsuccessful and very successful steps need to be adapted. Nevertheless, for the sake of completeness, we provide a self contained analysis adapted to our notation.

Lemma 4.1 *$\|d_k\| \geq \kappa_g\sqrt{\|\nabla f(x_{k+1})\|}$ for all successful iterations k, where*

$$\kappa_g \stackrel{\text{def}}{=} \sqrt{\frac{1}{\frac{1}{2}L_H + L_0}}.$$

Proof. Denoting $g_{k+1} = g(x_k + d_k) = \nabla f(x_k + d_k)$ and g_k accordingly, we use a generalization of the fundamental theorem of integral calculus [21, 8.1.2] to write

$$g_{k+1} = g_k + \int_0^1 d_k^t H(x_k + \tau d_k) d\tau$$

On the other hand, d_k being a global minimizer of $m_k^{\alpha_k}$, equation (5) yields

$$\nabla m_k^{\alpha_k}(d_k) = g_k + d_k(H(x_k) + \lambda_k I) = 0$$

so that

$$\begin{aligned}
\|g_{k+1}\| = \|g_{k+1} - \nabla m_k^{\alpha_k}(d_k)\| &= \left\| \left(\int_0^1 d_k^t H(x_k + \tau d_k) d\tau \right) - d_k^t (H(x_k) + \lambda_k I) \right\| \\
&= \left\| \left(\int_0^1 d_k^t (H(x_k + \tau d_k) - H(x_k)) d\tau \right) - \lambda_k d_k^t \right\| \\
&\leq \|d_k\| \left\| \left(\int_0^1 L_H \tau d_k d\tau \right) \right\| + \|\lambda_k d_k\| \\
&\leq \left(\frac{L_H}{2} + \frac{1}{\alpha_k} \right) \|d_k\|^2 \leq \left(\frac{L_H}{2} + L_0 \right) \|d_k\|^2
\end{aligned}$$

■

The above bound does not hold for the trust region step, lacking any information on the λ_k parameter. Thus, for trust region, we have the less precise bound $\|g_{k+1}\| \leq \left(\frac{L_H}{2}\|d_k\| + \lambda_k\right)\|d_k\|$.

We are now interested in bounding the total work the algorithm needs to reach a first iteration $k(\epsilon) < \infty$ for which $\|\nabla f(x_{k(\epsilon)})\| \leq \epsilon$. The global convergence analysis ensures that there exists such a $k(\epsilon) < \infty$. Therefore, for all the considered iterations before k, $\nabla f(x_j) > \epsilon, \forall j < k(\epsilon)$. Similarly, in this context, the parameter β_j will be bounded away from zero for $j \leq k(\epsilon)$. Actually, the global convergence analysis above provides hints to the important relations involved in the complexity analysis.

Following [6], let us note

$$\mathcal{S} \overset{\text{def}}{=} \{k \geq 0 : \text{iteration } k \text{ is successful or very successful}\}$$

and $\mathcal{S}_j \overset{\text{def}}{=} \{k \leq j : k \in \mathcal{S}\}$ and $\mathcal{U}_j \overset{\text{def}}{=} \{0 \leq k \leq j : k \notin \mathcal{S}\}$, i.e. iteration k is unsuccessful. Also, for sets such as $\mathcal{U}_j$, $|\mathcal{U}_j|$ denotes their cardinality.

Theorem 4.2 (Complexity bound of ARC$_q$) *The maximum number of successful iterations of ARC$_q$ is $|\mathcal{S}_j| \leq \frac{L_0}{\eta_1 \kappa_g^3 \epsilon^{\frac{3}{2}}} \left(f(x_0) - f(x_{low})\right) = L^s \epsilon^{-\frac{3}{2}}$. The maximum number of successful and unsuccessful iterations is*

$$|\mathcal{S}_j| + |\mathcal{U}_j| \leq \epsilon^{-\frac{3}{2}} \frac{\left(2 \log \gamma_3 L^s - \log\left(\frac{\alpha_0}{\bar{\alpha}}\right)\right)}{-\log \gamma_1} \tag{8}$$

where $\bar{\alpha} = \frac{1}{L_0}$ from Lemma 2.2.

Proof. For any $k < k(\epsilon)$, $\nabla f(x_k) > \epsilon$. The Lemmas 2.1 and 4.1 combine to obtain

$$f(x_k) - q_k(d_k) \geq \frac{\kappa_g^3}{L_0} \epsilon^{\frac{3}{2}}. \tag{9}$$

For successful iterates, $\frac{f(x_k) - f(x_{k+1})}{f(x_k) - q_k(d_k)} \geq \eta_1$ so that

$$f(x_k) - f(x_{k+1}) \geq \eta_1 (f(x_k) - q_k(d_k)) \geq \eta_1 \frac{\kappa_g}{L_0} \epsilon^{\frac{3}{2}}.$$

The rest of the analysis is identical to [6] and is given below for completeness.

Summing over all successful iterates before $k(\epsilon)$, and assuming that the monotonically decreasing sequence $\{f(x_k)\}$ is bounded below by f_{low}, we get (for $j = k(\epsilon)$)

$$f(x_0) - f_{\text{low}} \geq \sum_{k \in \mathcal{S}_j} (f(x_k) - f(x_{k+1})) \geq |\mathcal{S}_j| \eta_1 \frac{\kappa_g^3}{L_0} \epsilon^{\frac{3}{2}},$$

which we use to bound $|\mathcal{S}_j| \leq \frac{L_0}{\eta_1 \kappa_g^3 \epsilon^{\frac{3}{2}}} \left(f(x_0) - f(x_{\text{low}})\right) = L^s \epsilon^{-\frac{3}{2}}$.

To bound the number of unsuccessful iterations, note that the algorithm flow ensures that

$$\gamma_3 \alpha_k \geq \alpha_{k+1}, \forall k \in \mathcal{S}_j$$

and

$$\gamma_1 \alpha_i \geq \alpha_{i+1}, \forall i \in \mathcal{U}_j.$$

Therefore, $\alpha_0 \gamma_3^{|\mathcal{S}_j|} \gamma_1^{|\mathcal{U}_j|} \geq \alpha_j$ so that $|\mathcal{S}_j| \log \gamma_3 + |\mathcal{U}_j| \log \gamma_1 \geq \log\left(\frac{\bar{\alpha}}{\alpha_0}\right)$ which yields ($\gamma_1 < 1$)

$$|\mathcal{U}_j| \leq \frac{1}{\log \gamma_1} \left\lceil -\log \gamma_3 |\mathcal{S}_j| + \log\left(\frac{\alpha_0}{\bar{\alpha}}\right) \right\rceil \leq \left\lceil \epsilon^{-\frac{3}{2}} \left(\frac{\log \gamma_3}{-\log \gamma_1} L^s + \frac{1}{-\log \gamma_1} \epsilon^{\frac{3}{2}} \log\left(\frac{\alpha_0}{\bar{\alpha}}\right) \right) \right\rceil \tag{10}$$

so that for $\epsilon < 1$, the total number of iterations, both successful and unsuccessful is given by (8). ∎

A similar analysis for the trust region variant establishes that the worst case number of function evaluations is at best $\mathcal{O}(\epsilon^{-2})$. Lemma 2.1 for the trust region case implies a choice (the min) and if the min is always reached for $\frac{\|\nabla f(x_k)\|}{\|\nabla^2 f(x_k)\|}$, we already get that the relevant equation (9) becomes $f(x_k) - q_k(d_k) \geq \frac{1}{2M_H}\epsilon^2$ where $M_H \geq \|\nabla^2 f(x)\|$ and the remaining of the proof adapts to get the $\mathcal{O}(\epsilon^{-2})$ bound. We remark that this bound is not a complete worst case analysis since the other choice for the min could lead to even worse bounds, but this is not the case and the $\mathcal{O}(\epsilon^{-2})$ complexity bound is proved in [13]. The case outlined above is confirmed by an example reaching this complexity in [4], the example being such that the trust region is always inactive, thus the min is always reached by the $\frac{\|\nabla f(x_k)\|}{\|\nabla^2 f(x_k)\|}$ term. A recent (and rather complicated) variant of the trust region algorithms actually achieves the optimal complexity bound $\mathcal{O}(\epsilon^{-\frac{3}{2}})$ [9].

5. Efficient matrix implementation

A way to obtain a global minimizer for ARC is sketched in algorithm 6.1 in [5] and involves a Newton search on the parameter λ whose iterations' main computational work is to factorize the matrix $B(\lambda) = \nabla^2 f(x) + \lambda I$. This is the adaptation of the so called Moré-Sorenson [18] approach used in trust region algorithms.

In the algorithmic framework 1, each unsuccessful iteration requires again such a factorization. In the spirit of [11], we now propose to use a modified Cholesky factorization proposed in [7] to ensure that only one factorization is required for each successful step.

We intend to use the Bounded Bunch-Kauffman (BBK) pivoting strategy [7] to compute $P\nabla^2 f(x)P^t = LDL^t$. Next, we use a change of variables $\tilde{d} = L^t d$ and $\tilde{g} = gL^{-t}$. This opens a few possibilities, the most obvious being to use the cubic regularization term $\frac{\|d\|_M^3}{3}$ for the norm induced by the symmetric matrix $M = LL^t$. In [7] it is justified that the matrix $M = LL^t$ obtained by the BBK pivoting strategy is uniformly bounded and bounded away from zero. Another strategy uses the spectral factorization $H = Q^t\Lambda Q$ where Q is the orthogonal matrix of eigenvectors of H and Λ the diagonal matrix of eigenvalues.

Scaled regularization. If we adopt the $\|\cdot\|_M$ norm, the cubic model $c_x^\alpha(d) = q_x(d) + \frac{\|d\|_M^3}{3\alpha_k}$, $\|d\|_M = \sqrt{d^t LL^t d} = \|L^t d\|_2 = \|\tilde{d}\|$. We then get $\nabla c_x^\alpha(d) = \nabla q_x(d) + \frac{\|d\|_M}{\alpha} d^t M = g + d^t(H + \frac{\|\tilde{d}\|}{\alpha}M)$, so that applying the change of variables yields that $\nabla c_x^\alpha(d) = 0$ is equivalently solved with $\tilde{g} + \tilde{d}^t(D + \frac{\|\tilde{d}\|}{\alpha}I) = 0$ which is best done by searching $\lambda^* \approx \frac{\|\tilde{d}\|}{\alpha}$ such that $\tilde{g} + \tilde{d}^t(D + \lambda^* I) = 0$. The search for a suitable value for the λ parameter involves a block diagonal linear system.

Absolute value regularization. A slight variation is to use the second order information by defining $\hat{M} = L|D|L^t$ as suggested in [11]. We then adopt the $\|\cdot\|_M$ norm, the cubic model $c_x^\alpha(d) = q_x(d) + \frac{\|d\|_M^3}{3\alpha_k}$, $\|d\|_M = \sqrt{d^t L|D|L^t d} = \left\|\sqrt{|D|}L^t d\right\|_2 = \|\tilde{d}\|_{|D|}$. We then get $\nabla c_x^\alpha(d) = \nabla q_x(d) + \frac{\|d\|_M}{\alpha} d^t M = g + d^t(H + \frac{\|\tilde{d}\|_{|D|}}{\alpha}M)$, so that applying the change of variables yields that $\nabla c_x^\alpha(d) = 0$ is equivalently solved with $\tilde{g} + \tilde{d}^t(D + \frac{\|\tilde{d}\|_{|D|}}{\alpha}|D|) = 0$ which is best done by searching $\lambda^* \approx \frac{\|\tilde{d}\|_{|D|}}{\alpha}$ such that $\tilde{g} + \tilde{d}^t(D + \lambda^*|D|) = 0$. Here also the search for a suitable value for the λ parameter involves a block diagonal linear system.

Spectral factorization. Using the spectral factorization makes sense only for dense problems since nothing ensures that for a sparse matrix H, its eigenvectors would be sparse. In this case, the matrix $M = Q^t Q = I$ is obviously bounded and bounded away from zero. The spectral factorization actually corresponds to the using the usual euclidean norm to define the trust region or the regularization term since $M = I$. The spectral factorization may be combined with the absolute value scaling.

```
Scaled_ARC$_q(x, \alpha, f)$
  { Given: (initial) $x$, $\beta > 0$; }
  {        parameters $0 < \gamma_1 < \gamma_2 \leq 1 < \gamma_3$; }
  {        parameters $0 < \eta_1 < \eta_2 < 1$; }
  {        objective function $f$ and model $m$. }
  while ( $\neg$ termination_criterion )
    Factors $\leftarrow$ factorization($\nabla^2 f(x)$)
    success $\leftarrow$ false
    repeat
      $d \leftarrow$ Solve_Model($\alpha$, Factors, $\nabla f(x)$)
      $\Delta f \leftarrow f(x) - f(x+d)$
      $\Delta q \leftarrow q(0) - q(d)$
      $r \leftarrow \frac{\Delta f}{\Delta q}$
      if ($r < \eta_1$) then $\alpha \leftarrow \gamma_1 \alpha$   {Unsuccessful}
      else
        success $\leftarrow$ true
        $x \leftarrow x + d$                                   {Successful}
        if ($r > \eta_2$) then
          $\alpha \leftarrow \gamma_3 \alpha$                  {Very successful}
    until ( success)
  Result $\leftarrow x$
```

Algorithm 2: Scaled ARC$_q$.

In the algorithm 2, the inner repeat never involves $\mathcal{O}(n^3)$ matrix operations, so as claimed, only one factorization is necessary per successful iteration. Observe that no factorization is required for unsuccessful iterations and that `Solve_Model` will use the factorization to implement a safeguarded Newton iteration for computing the Levenberg-Marquardt parameter λ. Therefore, the only difference between the ARC$_q$ variant and the trust region is in the inner safeguarded Newton iterations to compute λ within the `Solve_Model` function.

5.1 *Some numerical comparisons*

We developed a unified implementation which uses any factorization and which may be combined with the absolute value scaling. Our implementation is done in the Julia language [2]. We consider the BBK and the spectral factorizations. The BBK factorization was adapted in Julia from Higham's Matlab code [15, ldlt_symm.m] while we use the spectral factorization available within Julia (from Lapack). We consider eight variants, sharing the bulk of the implementation: a trust region strategy and the ARC$_q$ strategy using either the LDLt or the Spectral factorization, with or without absolute value scaling.

For each factorization, we present a performance profile of the four variants (trust region or ARC$_q$, absolute value or not) on small instances (dimension 100) from the Lukšan *et al* [17] collection. The problems are coded in the JuMP [16] modelization language within Julia. We limit ourselves to small instances since neither the code for the BBK factorization nor the spectral factorization exploit sparsity.

The implementation uses $\eta_1 = 0.1$, $\eta_2 = 0.75$, $\gamma_1 = 0.1$, $\gamma_2 = 1$ and $\gamma_3 = 5$.

As observed in figures 1 and 2, the ARC$_q$ variants appear more robust. It is remarkable that both versions of ARC$_q$ with the absolute value scaling actually solve all problems of the

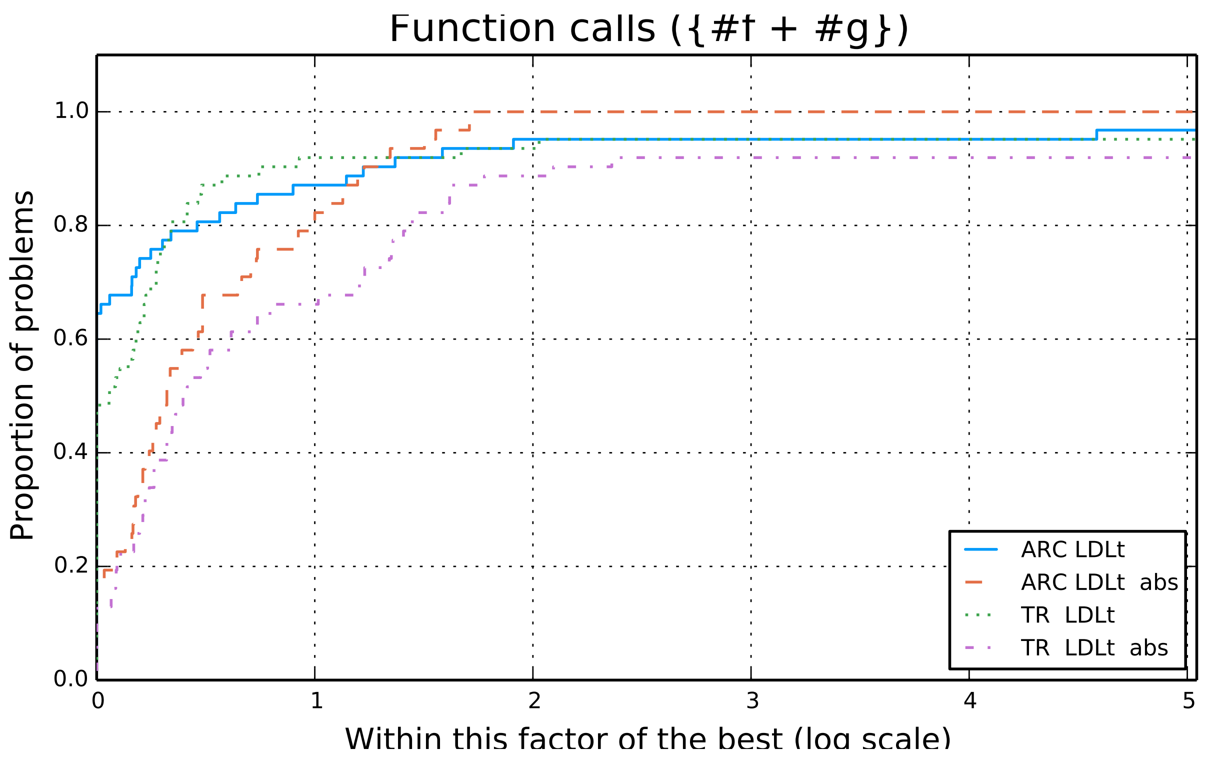

Figure 1. Solution of 60 examples from the Lukšan collection in dimension $n = 100$. The maximum iteration is set to 10000. The two ARC variant using the absolute value scaling solves all the test problems to the required tolerance $\|\nabla f(x_k\| \leq \max\{10^{-5}, 10^{-10}\|\nabla f(x_0)\|\}$.

collection. Efficiency appears on par with the TR variants. In figure 3, we may observe that the use of the scaled regularization by $M = LL^t$ yields similar performances as the usual L_2 norm as computed by the spectral factorization.

5.2 *Some large scale sparse examples*

We present preliminary experimentations to assess the scaling potential of the method. We use the MA97 factorization from the HSL library [1], a FORTRAN code able to factorize sparse matrices efficiently. This software does not seem to ensure that $\|L\|$ remains bounded and bounded away from zero, so the robustness of this preliminary code is not as high as we would expect.

To assess the practical efficiency of ARC$_q$, we compare its performance to the well known L-BFGS-B code. Since both methods do not use the same information (second order), it is difficult to bypass the use of CPU timing for the comparisons. We observe that ARC$_q$ is less efficient but nevertheless competitive with L-BFGS-B but even for the MA97 factorization (which does not ensure that $\|L\|$ remains bounded and bounded away from zero), is significantly more robust than L-BFGS-B.

Conclusion

We have introduced a new variant ARC$_q$ of the adaptive regularization by cubics which is very closely related to trust region schemes. We provided a unified convergence proof both simple and intuitive. Among other, we observed that the steps in both methods are given by the solution

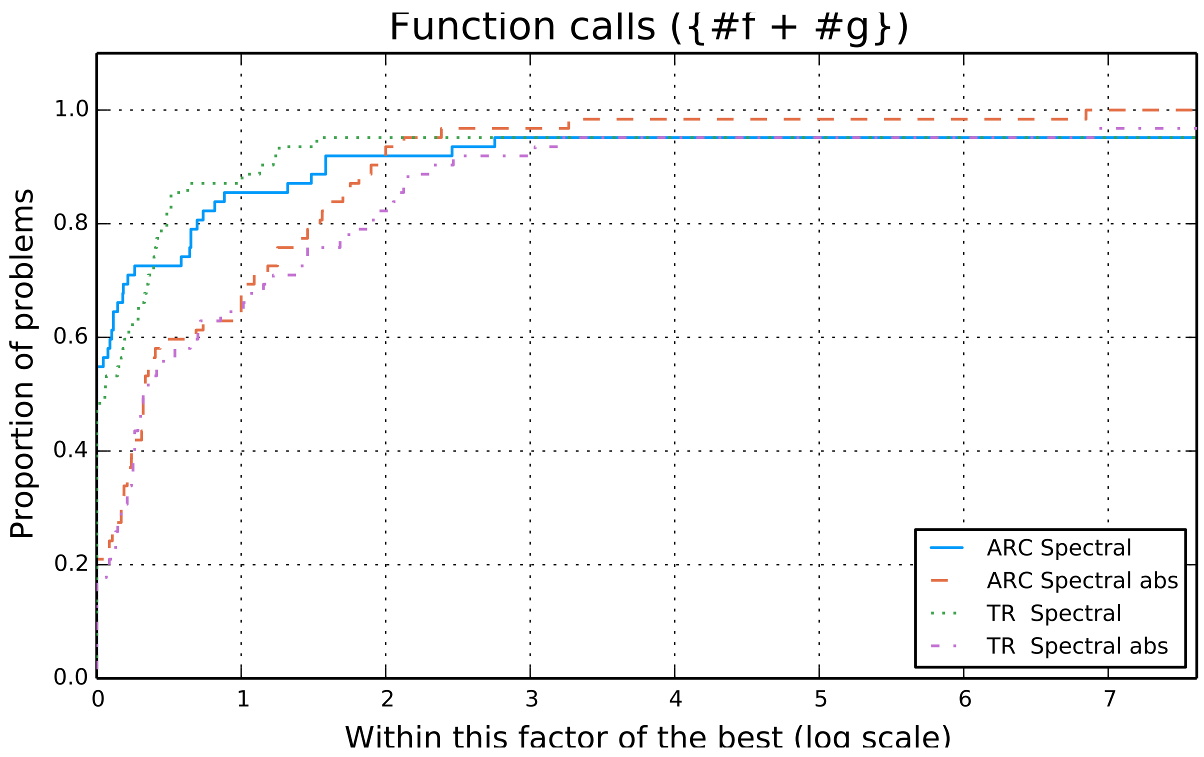

Figure 2. Solution of 60 examples from the Lukšan collection in dimension $n = 100$. The maximum iteration is set to 10000. The Spectral variant with absolute value scaling solves all the test problems to the required tolerance $\|\nabla f(x_k\| \leq \max\{10^{-5}, 10^{-10}\|\nabla f(x_0)\|\}$.

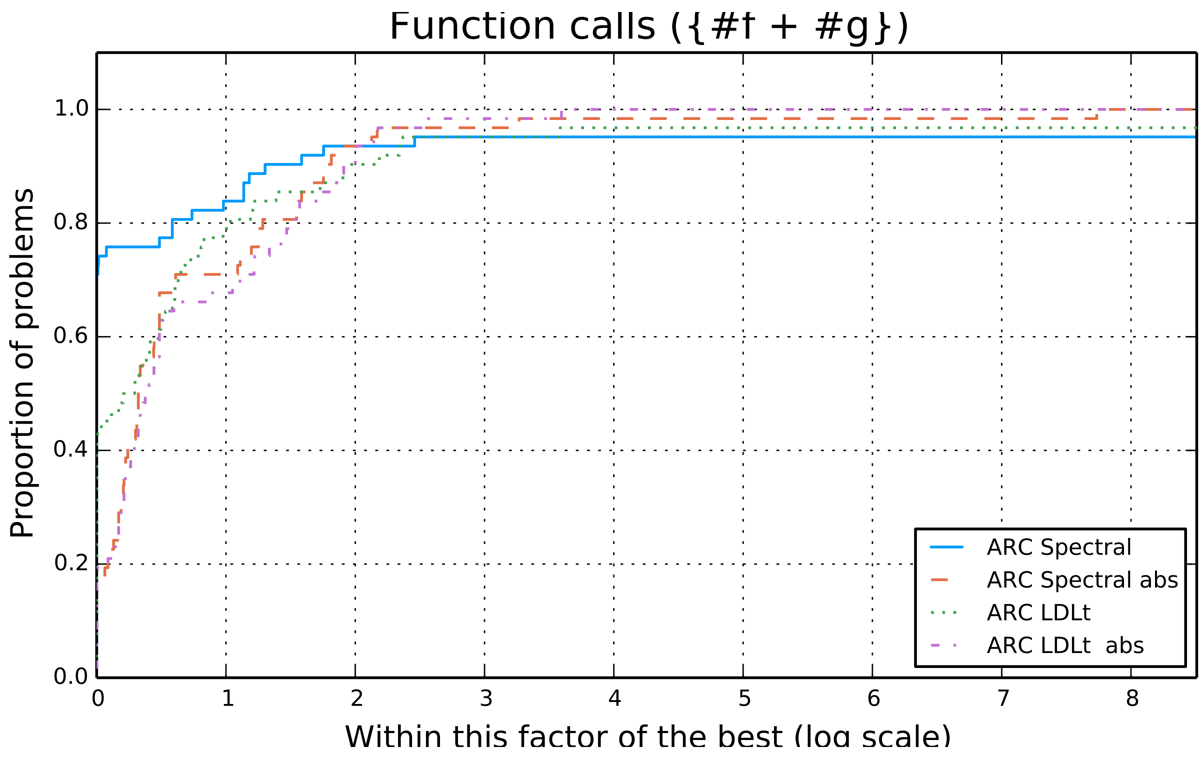

Figure 3. Solution of 60 examples from the Lukšan collection in dimension $n = 100$. The maximum iteration is set to 10000 and the stopping tolerance is $\|\nabla f(x_k\| \leq \max\{10^{-5}, 10^{-10}\|\nabla f(x_0)\|\}$. The LDLt variants appear competitive with the Spectral ones, confirming that the use of the scaling induced by $M = LL^t$ is not detrimental with respect to the usual L_2 norm

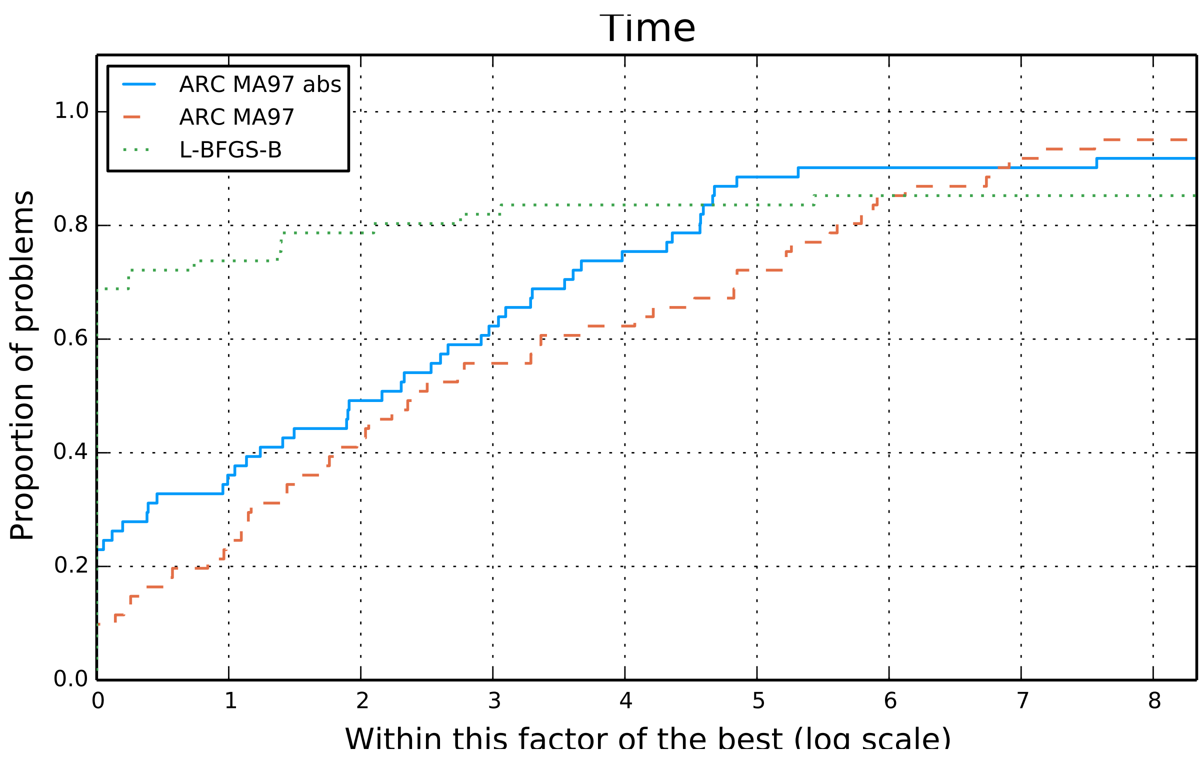

Figure 4. Solution of 60 examples from the Lukšan collection in dimension $n = 10000$. The maximum iteration is set to 10000, the stopping tolerance is $\|\nabla f(x_k\| \leq \max\{10^{-5}, 10^{-10}\|\nabla f(x_0)\|\}$.

of a regularized linear system of the Levenberg-Marquardt type

$$\left(\nabla^2 f(x_k) + \lambda_k I\right) d_k = -\nabla f(x_k),$$

differing only by the λ parameter used.

Our analysis showed that the optimal complexity property relies on the fact that $\lambda_k \sim \Omega(\|d_k\|)$ in Lemma 2.1 and that $\lambda_k \sim \mathcal{O}(\|d_k\|)$ in Lemma 4.1, so that things work extremely well if d_k is computed as a global minimiser of the cubic model in which case $\lambda_k = \frac{\|d_k\|}{\alpha_k}$.

We finally provided an efficient matrix implementation avoiding any matrix factorization for unsuccessful iterations. This implementation uses a scaling which was shown non-detrimental with respect to the usual L_2 norm in such algorithms, the figure 3 clearly exhibiting equivalent efficiency and robustness for the LDLt and spectral variants.

Future work will concern applicability to large scale problems, including further investigation of BBK factorizations exploiting sparsity and the development of iterative methods to solve the cubic regularized subproblems.

References

[1] HSL. a collection of Fortran codes for large scale scientific computation. http://www.hsl.rl.ac.uk/.

[2] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B. Shah. Julia: A fresh approach to numerical computing. November 2014.

[3] Ernesto Birgin, John Gardenghi, Jos-Mario Martinez, Sandra Augusta Santos, and Philippe L. Toint. Worst-case evaluation complexity for unconstrained nonlinear optimization using high-order regularized models. Workingpaper ¡importmodel: Workingpaperimportmodel¿, Namur center for complex systems, 6 2015.

[4] Coralia Cartis, Nicholas I.M. Gould, and Philippe L. Toint. On the complexity of steepest descent, newton's and regularized newton's methods for nonconvex unconstrained optimization problems. *SIAM Journal on Optimization*, 20(6):2833–2852, 2010.

[5] Coralia Cartis, Nicholas I.M. Gould, and Philippe L. Toint. Adaptive cubic regularisation methods for unconstrained optimization. Part I: motivation, convergence and numerical results. *Mathematical Programming*, 127(2):245–295, 2011.

[6] Coralia Cartis, Nicholas I.M. Gould, and Philippe L. Toint. Adaptive cubic regularisation methods for unconstrained optimization. Part II: worst-case function- and derivative-evaluation complexity. *Mathematical Programming*, 130(2):295–319, 2011.

[7] Sheung Hun Cheng and Nicholas J. Higham. A modified Cholesky algorithm based on a symmetric indefinite factorization. *SIAM J.*MATRIX ANAL. APPL., 19(4):1097–1110, October 1998.

[8] Andrew R. Conn, Nicholas I.M. Gould, and Philippe L. Toint. *Trust Region Methods*. MPS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, 2000.

[9] Frank E. Curtis, Daniel P. Robinson, and Mohammadreza Samadi. A trust region algorithm with a worst-case iteration complexity of $\wr(\epsilon^{-3/2})$ for nonconvex optimization. *Mathematical Programming*, pages 1–32, 2016.

[10] Roger Fletcher. *Practical Methods of Optimization; (*2^{nd} *Ed.)*. Wiley-Interscience, New York, NY, USA, 1987.

[11] Nicholas I.M. Gould and Jorge Nocedal. The modified absolute-value factorization norm for trust-region minimization. In Renato De Leone, Almerico Murli, PanosM. Pardalos, and Gerardo Toraldo, editors, *High Performance Algorithms and Software in Nonlinear Optimization*, volume 24 of *Applied Optimization*, pages 225–241. Springer US, 1998.

[12] Geovani N. Grapiglia, Jinyun Yuan, and Ya-xiang Yuan. On the convergence and worst-case complexity of trust-region and regularization methods for unconstrained optimization. *Mathematical Programming*, 152(1):491–520, 2015.

[13] Serge Gratton, Annick Sartenaer, and Philippe L. Toint. Recursive trust-region methods for multiscale nonlinear optimization. *SIAM Journal on Optimization*, 19(1):414–444, 2008.

[14] Andreas Griewank. The modification of Newtons 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] Nicholas J. Higham. The Matrix Computation Toolbox. `http://www.ma.man.ac.uk/~higham/mctoolbox`.

[16] Miles Lubin and Iain Dunning. Computing in operations research using julia. *INFORMS Journal on Computing*, 27(2):238–248, 2015.

[17] Ladislav Lukšan, Ctirad Matonoha, and Jan Vlček. Modified CUTE problems for sparse unconstrained optimization. Technical Report 1081, Institute of Computer Science, Academy of Science of the Czech Republic, October 2010. http://www.cs.cas.cz/matonoha/download/V1081.pdf.

[18] Jorge J. Moré and Danny C. Sorensen. Computing a trust region step. *SIAM Journal on Scientific and Statistical Computing*, 4(3):553–572, 1983.

[19] Yurii Nesterov and Boris T. Polyak. Cubic regularization of newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006.

[20] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006.

[21] James M. Ortega. *Numerical analysis: A second course.* Philadelphia, PA: SIAM, 1990.

[22] Philippe L. Toint. Nonlinear stepsize control, trust regions and regularizations for unconstrained optimization. *Optim. Methods Softw*, 2013.