A NON-MONOTONE TRUST-REGION ALGORITHM FOR NONLINEAR OPTIMIZATION SUBJECT TO CONVEX CONSTRAINTS

Philippe L. Toint

Report 94/24 November 14, 1994

Department of Mathematics
Facultés Universitaires ND de la Paix
B-5000 Namur, Belgium

Current reports available by anonymous ftp from the directory
`pub/reports` on thales.math.fundp.ac.be (internet 138.48.4.14).

Keywords: non-monotone algorithms, trust regions, convex constraints.

A non-monotone trust-region algorithm for nonlinear optimization subject to convex constraints

Philippe L. Toint

November 14, 1994

Abstract

This paper presents two new trust-region methods for solving nonlinear optimization problems over convex feasible domains. These methods are distinguished by the fact that they do not enforce strict monotonicity of the objective function values at successive iterates. The algorithms are proved to be convergent to critical points of the problem from any starting point. Extensive numerical experiments show that this approach is competitive with the LANCELOT package.

1 Introduction

In this paper, we consider algorithms for finding a local solution of the problem

$$\min_{x \in X} \quad f(x), \tag{1.1}$$

where $f(x)$ is a real-valued twice continuously differentiable function, which we assume is bounded below, and where X is a closed convex subset of $\mathbf{R}^n$. Because of its practical importance in its own right and as a common subproblem in more complex procedures, this problem has received a lot of attention from the research community. In particular, the last years have seen the development and implementation of several "trust-region methods" for its solution (see, for instance, Burke and Moré (1988), Burke *et al.* (1990), Calamai and Moré (1987), Conn *et al.* (1988), Conn *et al.* (1993) and Toint (1988)). Trust-region methods are iterative numerical algorithms in which the objective function $f(x)$ is approximated in a suitable neighbourhood of the current iterate (the *trust region*) by a model whose values are easy to compute. A step is then chosen in the intersection of this neighbourhood and X that sufficiently reduces the value of this model. The true objective function is computed at the resulting trial point: if it has also decreased sufficiently, the trial point is accepted as the new iterate. The trust region is then increased if the objective function value was well predicted by the model, or decreased otherwise.

Most trust-region algorithms proposed up to now are *descent methods*, in that they only accept the trial point as next iterate if its associated objective function value is strictly lower than that at the current iterate. This monotonicity property ensures that each "successful iteration" produces a point that is better than any other point found so far, a property which is heavily used in the theoretical justifications for such algorithms. It is the purpose of this paper to explore the possibility of abandoning this algorithmic restriction. We will thus introduce, analyze and experiment a trust-region method for the solution of problem (1.1) such that the sequence $\{f(x_k)\}$ of function values at the k-th iterate x_k is no longer monotone.

The idea of non-monotone local optimization methods is not new, but has not yet been explored to any large extent. Perhaps the first such method is the "watchdog technique" introduced by Chamberlain *et al.* (1982) in order to avoid the so-called Maratos effect in certain constrained optimization methods. This idea was developed further by Grippo, Lampariello and Lucidi in a few seminal papers (see Grippo *et al.* (1986) and Grippo *et al.* (1991)) where they investigated linesearch methods for unconstrained problems. Their conclusions were overall very favourable to non-monotone methods, especially when applied to very nonlinear problems. These conclusions were recently revisited by Toint (1994a), where it was shown that these methods may often provide subtantial efficiency gains. Other references about non-monotone linesearch techniques include Panier and Tits (1991), Bonnans *et al.* (1992) and Facchinei and Lucidi (1993). The work reported on in the current paper resulted from the natural question of extending the idea from linesearch algorithms to trust-region methods. Again, this idea has already been explored to some extent by Xiao and Zhou (1992) and Deng *et al.* (1993), although from a rather different point of view.

The paper is organized as follows. Section 2 introduces the classical monotone trust-region framework, the algorithm by Deng, Xiao and Zhou and a first new non-monotone variant in detail. Section 3 presents the results of extensive numerical experience, where the new method is compared to the LANCELOT package of Conn *et al.* (1992) and to the proposal of Deng, Xiao and Zhou. A further improvement of the proposed algorithm is introduced in Section 4 and numerically experimented in Section 5. Conclusions are finally discussed in Section 6. The appendix presents the global convergence theory for both new algorithms.

2 Three algorithms

We first introduce some notation. We denote by $\langle \cdot, \cdot \rangle$ the usual Euclidean inner product of $\mathbf{R}^n$, and by $\| \cdot \|$ its associated norm. The sequence of iterates generated by the discussed algorithms is denoted by $\{x_k\}$ and we define, for all k, $g_k = \nabla f(x_k)$.

Because the method we wish to discuss is a variant of the (now) standard trust-region methodology, we first describe this framework. We thus consider an iterative technique producing a sequence $\{x_k\}$ hopefully converging to a critical point of problem (1.1), that is a point x_* such that

$$-\nabla f(x_*) \in N(x_*), \tag{2.1}$$

where $N(x_*)$ is the normal cone to X at x_*. The proximity of an iterate x_k of a critical point is measured by a suitable criticality measure $\alpha_k = \alpha(x_k)$ which we assume is continuous, strictly positive at all points that are not critical and zero at critical points. For instance, the definition

$$\alpha(x) = \|P_X(x - \nabla f(x)) - x\|^2 \tag{2.2}$$

is used by Toint (1988), where $P_X(\cdot)$ is the orthogonal projection onto X. Other choices are possible (see Conn *et al.* (1988) and Conn *et al.* (1993), for instance).

At each iterate x_k, we define a model $m_k(x)$ whose purpose is to approximate the objective function in the trust region given by

$$B_k = \{x \in \mathbf{R}^n \mid \|x - x_k\| \le \Delta_k\}, \tag{2.3}$$

where Δ_k is called the trust-region radius. For simplicity of exposition, we assume that, for all k, m_k is twice differentiable on $B_k \cap X$,

$$m_k(x_k) = f(x_k), \quad \nabla m_k(x_k) = g_k \text{ and } \|\nabla^2 m_k(x)\| \le \omega - 1 \text{ for all } x \in B_k \cap X, \tag{2.4}$$

where $\omega \ge 1$ is a constant. A step s_k to a feasible trial point is then sought that ensures sufficient reduction in this model in view of the value of α_k and the trust region radius. More specifically, we require that $x_k + s_k$, the trial point, belongs to $B_k \cap X$ and that

$$m_k(x_k) - m_k(x_k + s_k) \ge \kappa \alpha_k \min\left[\frac{\alpha_k}{b_k}, \Delta_k\right], \tag{2.5}$$

for some constant $\kappa \in (0, 1)$, where

$$b_k = 1 + \max_{x \in B_k \cap X} \|\nabla^2 m_k(x)\|. \tag{2.6}$$

Note that our assumption on the second derivatives of the model implies that

$$b_k \le \omega. \tag{2.7}$$

The achievability of (2.5) for suitable choices of α_k, has been shown by several authors (see Conn *et al.* (1988), Conn *et al.* (1993), Moré (1988) and Toint (1988), for instance).

2.1 The basic trust-region framework

We are now in position to define our standard trust-region algorithm more formally.

Basic trust-region Algorithm (BTR)

Step 0: Initialization. An initial point x_0 and an initial trust-region radius Δ_0 are given. The constants η_1, η_2, γ_1, γ_2 and γ_3 are also given and satisfy the conditions

$$0 < \eta_1 \le \eta_2 < 1 \text{ and } 0 < \gamma_1 \le \gamma_2 < 1 \le \gamma_3. \tag{2.8}$$

Set $k = 0$.

Step 1: Model definition. Define a model m_k satisfying conditions (2.4).

Step 2: Step calculation. Compute a step s_k such that $x_k + s_k \in B_k \cap X$ and (2.5) holds.

Step 3: Acceptance of the trial point. Compute $f(x_k + s_k)$ and define the ratio

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}. \tag{2.9}$$

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

Step 4: Trust-region radius update. Set

$$\Delta_{k+1} \in \begin{cases} [\Delta_k, \gamma_3 \Delta_k] & \text{if } \rho_k \geq \eta_2, \\ [\gamma_2 \Delta_k, \Delta_k] & \text{if } \rho_k \in (\eta_1, \eta_2), \\ [\gamma_1 \Delta_k, \gamma_2 \Delta_k] & \text{if } \rho_k \leq \eta_1. \end{cases} \tag{2.10}$$

Increment k by one and go to Step 1.

This framework is quite general and leaves many details unspecified. For instance, the choice of a suitable model m_k is not detailed in Step 1, where one would typically choose, for a practical implementation, a model of the form

$$m_k(x_k + s) = f(x_k) + \langle g_k, s \rangle + \frac{1}{2} \langle s, H_k s \rangle, \tag{2.11}$$

where H_k is a symmetric bounded approximation to $\nabla^2 f(x_k)$. The precise way in which the step s_k is computed is not described either at this stage. However, the present description is sufficient to introduce the non-monotone algorithm, and some additional details will be discussed further in relation with our practical implementation (see Section 3). As is usual, we call an iteration "successful" whenever $\rho_k \geq \eta_1$, and "unsuccessful" otherwise. We denote by $\mathcal{S}$ the subset of indices of all successful iterations.

2.2 The algorithm of Deng, Xiao and Zhou

The idea of non-monotone techniques is to examine the performance of the algorithm more than one iteration at a time to determine if a particular trial point should be accepted or rejected. We now turn to the description of a previous proposal for a non-monotone trust-region method due to Xiao and Zhou (1992) and Deng *et al.* (1993). Since the first of these papers uses a framework similar to the second and is more complicated, we will focus here on the description presented in the second.

The main proposal of Deng and colleagues may be viewed as a modification of Step 3 of Algorithm BTR. In order to describe this modification, we use the counter ℓ (initialized to 0 in Step 0) to index the sequence of successful iterations and the sequence $\{f_\ell\}$ is the sequence of objective function's values at successful iterations. The algorithm also depends on the real constants $\mu \in (0, 1)$ and $\gamma_4 > 0$ and on a positive integer p representing the number of past successful iterations one is willing to consider when accepting or rejecting a new iterate.

Step 3 of the trust-region algorithm by Deng, Xiao and Zhou (DXZ)

Step 3: Acceptance of the trial point. If $\ell > 0$, define an index $m(k)$ such that

$$f_{m(k)} = \max_{i=\max[\ell-p+1,1],\ldots,\ell} f_i \tag{2.12}$$

and compute

$$\mu_k = \min\left[\frac{f_{m(k)} - f(x_k) - \gamma_4\alpha_k\Delta_k}{m_k(x_k+s_k) - m_k(x_k)}, \mu\right] \tag{2.13}$$

and

$$\rho_k = \frac{f(x_k) - f(x_k+s_k)}{m_k(x_k) - m_k(x_k+s_k)}. \tag{2.14}$$

If $\rho_k < \mu_k$, then define $x_{k+1} = x_k$. Otherwise, define $x_{k+1} = x_k + s_k$, increment ℓ by one and set

$$f_\ell = f(x_k). \tag{2.15}$$

In their paper, Deng, Xiao and Zhou only consider purely unconstrained problems. They allow for the gradient $\nabla f(x_k)$ to be approximated by a vector g_k provided this approximation is asymptotically exact whenever the step norm tends to zero (see Moré (1983), Toint (1988) and Conn *et al.* (1993) for a similar schemes). They use the criticality measure $\alpha_k = \|g_k\|$, which is most natural choice in their framework. They also assume that the model m_k is quadratic and finally impose a maximal upper bound $\bar{\Delta}$ on the trust-region radii. (Surprisingly, the values suggested for this bound are between one and ten in their tests, and varies considerably from one problem to the next.)

The main idea behind the DXZ Algorithm is that either the minimum is acheived by the second term (μ) in (2.13) and we obtain a classical trust-region iteration where η_1 is replaced by μ, or the minimum is achieved by the first term of (2.13), and (2.13), (2.14) and the inequality $\rho_k \geq \mu_k$ together imply that, for a successful iteration,

$$f_{m(k)} - f(x_{k+1}) \geq \gamma_4\alpha_k\Delta_k \geq \gamma_4\alpha_k \min\left[\frac{\alpha_k}{b_k}, \Delta_k\right] \tag{2.16}$$

which is crucial for establishing a suitable convergence theory (see, for instance, the similar inequality (A.2) used in the appendix of this paper). This technique has the minor inconvenient to require an explicit value for μ (Deng, Xiao and Zhou suggest 0.001), as is the case in linesearch methods where one needs an explicit coefficient to define the upper bound on the admissible function value in the Wolfe, Goldstein or Powell's conditions (see Dennis and Schnabel (1983) or Fletcher (1987)).

Deng, Xiao and Zhou then prove global convergence for their algorithm, making use of (2.16) but also of an important additional requirement on the step s_k, namely that

$$\|s_k\| \leq \bar{\kappa}\alpha_k = \bar{\kappa}\|g_k\| \tag{2.17}$$

for some $\bar{\kappa} > 0$. They also analyze convergence of their method to local minimizers under the further assumption that negative curvature is correctly taken into account in the determination of the step s_k. They conclude their presentation with the presentation of a variant of the truncated conjugate-gradients method (see Steihaug (1983) and Toint (1981)) ensuring condition (2.17) and a few numerical tests indicating that the technique might be promising. Whether (2.17) is a reasonable condition is open to debate. Besides the need of a specialized algorithm for determining s_k, this condition prevents large steps in regions where the gradient is small, as in the neighbourhood of saddle points and at the bottom of valleys. This can be circumvented by chosing $\bar{\kappa}$ very large (Deng, Xiao and Zhou suggest a value of the order of 10^5 times the number of conjugate-gradient iterations required in the computation of s_k!), but this strategy indicates that (2.17) may not be essential.

Algorithm DXZ measures the "sufficient decrease" in terms of a linearized model (the quantity $\alpha_k \Delta_k$ in (2.13)). The adhoc character of this mixture between a general (possibly quadratic) model m_k and a specific linear model for the non-monotone strategy is the reason of our proposal, where the general model m_k is used throughout.

2.3 A new non-monotone trust-region algorithm

In order to enforce the coherence of the non-monotone trust-region approach, we thus propose to compute the value of ρ_k by comparing the achieved and predicted reductions (on the general model) not only for iteration k alone, but for the p last iterations. More specifically, we also propose to replace Step 3 by the procedure that we now describe.

In this procedure, we again use the index ℓ as a counter of the successful iterations. We assume that ℓ is initialized to zero in Step 0 of the algorithm, which means that "iteration zero" is considered as successful. We also use p and f_ℓ with the same meaning as in Algorithm DXZ. We finally define, at iteration k, the quantity

$$d_k = m_k(x_k) - m_k(x_k + s_k), \tag{2.18}$$

which is the model decrease at this iteration.

Step 3 of a non-monotone trust-region algorithm (NMTR1)

Step 3: Acceptance of the trial point. Compute the largest index $m(k)$ such that

$$f_{m(k)} = \max_{i=\max[\ell-p,0],\ldots,\ell} f_i. \tag{2.19}$$

Then compute $f(x_k + s_k)$ and the ratios

$$\rho_{1,k} = \begin{cases} \dfrac{f_{m(k)} - f(x_k + s_k)}{\sum_{i=m(k)}^{\ell} d_i} & \text{if } \ell > 0, \\ 0 & \text{otherwise,} \end{cases} \tag{2.20}$$

and

$$\rho_{2,k} = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}, \tag{2.21}$$

and set

$$\rho_k = \max[\rho_{1,k}, \rho_{2,k}]. \tag{2.22}$$

If $\rho_k < \eta_1$, then define $x_{k+1} = x_k$. Otherwise, define $x_{k+1} = x_k + s_k$, increment ℓ by one and set

$$f_\ell = f(x_k) \text{ and } d_\ell = m_k(x_k) - m_k(x_{k+1}). \tag{2.23}$$

We now briefly comment on this proposal.

1. We note that the value of ρ_k given by (2.22) is always larger or equal to that calculated by the standard procedure ($\rho_{2,k}$). Hence the effect of the new Step 3 is to accept steps that would be rejected by Algorithm BTR, provided they guarantee a sufficient decrease of the objective function over the past iterations containing the last p successful steps. The hope is that such steps might improve the efficiency of the minimization.

2. At variance with the standard trust-region scheme, the proposed non-monotone variant requires the storage of the last p terms of the sequences $\{f_\ell\}$ and $\{d_\ell\}$. As p is typically rather low, this additional storage is very acceptable. Similarly, the overhead in computing the new Step 3 instead of the standard one is also of order p, which can be considered as negligeable compared to the cost of computing the step s_k for instance, except maybe for very small problems.

We close this section by introducing some additional notation. For each $k \in \mathcal{S}$, we define

$$r(k) = \begin{cases} \text{the index of the } m(k)\text{-th successful iteration} & \\ \quad \text{in the complete sequence of all iterates} & \text{if } \rho_k = \rho_{1,k}, \\ k & \text{if } \rho_k = \rho_{2,k}, \end{cases} \tag{2.24}$$

that is the unique index, in the complete sequence of all iterates, of the successful iteration whose associated function value $f(x_{r(k)})$ is used in the definition of ρ_k from (2.22) and either (2.20) (with $f(x_{r(k)}) = f_{m(k)}$) or (2.21) with $f(x_{r(k)}) = f(x_k)$. We will call iteration $r(k)$ the *reference iteration* associated with iteration k.

The convergence theory for Algorithm NMTR1 is presented in Appendix.

3 Some numerical experiments with Algorithms BTR, DXZ and NMTR1

After introducing algorithms BTR, DXZ and NMTR1 in the previous section, we now report on numerical comparisons between a standard and two non-monotone trust-region methods. The standard trust-region algorithm chosen as yard-stick is the SBMIN routine, a component of the LANCELOT package of Conn *et al.* (1992). Since the details of this latter algorithm are described in detail in this reference, we will not cover them at any length. It suffices for the purpose of our comparison to note the following characteristics.

- It is a member of the general class of algorithms defined above as Algorithm BTR,

- The step s_k is computed using a truncated conjugate-gradient method to (approximately) minimize the model m_k within the trust region. (Note that Grippo *et al.* (1989) also use this technique, but, at variance with SBMIN, insist on an accurate minimization.)
- A variety of preconditioners is provided to improve the efficiency of the calculation of the step.
- The trust region is determined by using the ℓ_∞-norm in (2.3).

We then modified the SBMIN code to obtain Algorithm NMTR1, exactly following the description of Section 2.3. We also chose $p = 10$ and finally decided, after some experimenting, to apply the standard Step 3 of Algorithm BTR for the first 10 iterations.

We also modified SBMIN according to the algorithmic description of Section 2.2 to obtain a comparable version of Algorithm DXZ. The resulting code is not entirely equivalent to the method experimented by Deng, Xiao and Zhou, in that all the additional features of this method discussed in the text of Section 2.2 were not implemented. In particular, no special precaution was taken to enforce or verify condition (2.17). Despite these differences, it is the author's belief that the resulting code captures the main features of the proposal by Deng *et al.* According to the suggestion of these authors, the algorithm's constants

$$p = 10, \quad \mu = 0.7 \;\text{ and }\; \gamma_4 = 0.001 \tag{3.1}$$

were used in our implementation.

In a first set of tests, we then ran SBMIN, DXZ and NMTR1 on 327 unconstrained problems available[1] in the CUTE collection of test problems (see Bongartz *et al.* (1994)): 226 of these are "small problems" in that their dimension does not exceed 500, but 101 are larger with sizes from 500 to 16000. Furthermore, as SBMIN is used within LANCELOT as a subproblem solver for generally constrained problems, the same is also possible for DXZ and NMTR1. Since these subproblems also constitute interesting applications of the techniques explored here, we decided to add 63 difficult contrained test cases from CUTE to our test set, therefore containing 390 problems altogether. One should however keep in mind that what is tested in this context is not uniquely the reliability and efficiency of trust-region schemes, but their quality as a subproblem solver in the context of the augmented Lagrangian approach used in LANCELOT.

In our experiments, the default LANCELOT band preconditioner was used throughout. This preconditioner extracts the banded part of the true Hessian matrix (with semi-bandwith five) and uses a possibly modified factorization of this submatrix as preconditioner. The factorization is modified according to the technique proposed by Schnabel and Eskow (1991) when the banded submatrix is not positive definite. The reader is referred to Conn *et al.* (1992) for more details. All tests were performed on a DEC 3000/500 under OSF and using version 3.3 of the Fortran compiler with the default optimizing option. The criticality measure is $\alpha_k = \|g_k\|$ and convergence is assumed as soon as an iterate is found such that $\alpha_k \leq 10^{-5}$. Since the total number of test runs

[1] In March 1994.

is too large to report here in detail, we only present and discuss statistical summaries. However, the complete numerical results are available in a separate report (Toint (1994b))[2].

We first examine the relative reliabilities of SBMIN, DXZ and NMTR1, as shown in Table 1. We immediately note that NMTR1 is as robust as SBMIN, which is reassuring, given the excellent reliability of SBMIN. The reliability of DXZ is also very good.

	SBMIN	DXZ	NMTR1
successes	382	378	382
failures	8	12	8

Table 1: Reliability of SBMIN, DXZ and NMTR with band preconditioning on 390 problems

If we now consider all the problems solved, 373 were solved coherently, which means that all algorithms reported the same optimal value. The remaining 13 are excluded from our efficiency comparison between the three algorithms.

Table 2 shows the cumulative statistics for the three methods on these remaining 373 problems. In this table and subsequent ones, "iters" stands for the total number of iterations needed to solve all problems, "CG-its" for the total number of iterations within the conjugate-gradients procedure defining the search direction and "time" for the total CPU time (rounded to the second). Remember that there is one evaluation of the objective function per iteration. These cumulative results suggest that both NMTR1 and DXZ provide a significant average improvement on SBMIN for all efficiency measures and all problem sizes. On the other hand, the performance of DXZ appears to be poorer, at least on average.

	iters	CG-its	time
SBMIN	34073	500167	13616
DXZ	46614	706868	29275
NMTR1	26093	493276	11988

Table 2: Cumulative statistics for SBMIN, DXZ and NMTR1 with band preconditioning on 373 problems

In order to refine our conclusions, we now look at more disaggregate performance measures and report, in Table 3, how many times each of the three considered methods is best, second or worst. In establishing these rankings, two CPU times are reputed equal when they differ by less than 5% of the best one, or by less than half a second, or if they correspond to runs identical for all other measures. We also report, in Table 3, the *expected rank* for each method, which we

[2] This report may be obtained by anonymous `ftp` from the directory `pub/reports` on `thales.math.fundp.ac.be` or on simple request addressed to the author.

denote by e. This expected rank is given, for each algorithm, by

$$e = \frac{1}{n_t} \sum_{i=1}^{n_a} i n_i \tag{3.2}$$

where n_t is the total number of problems coherently solved by all algorithms (373 here), n_a is the number of algorithms (3 in our case) and n_i the number of times the considered algorithm is ranked in the i-th position, the lower rank corresponding to the better performance. This quantity will not directly be commented on in the results that follow, but it is the author's opinion that it represents a good synthetic measure of the relative merits of the n_a algorithms considered.

	method	1st	2nd	3rd	e
iters	SBMIN	257	51	65	1.49
	DXZ	270	68	35	1.37
	NMTR1	275	83	15	1.30
CG-its	SBMIN	256	56	61	1.48
	DXZ	262	73	38	1.40
	NMTR1	273	82	18	1.32
time	SBMIN	276	60	37	1.36
	DXZ	315	44	14	1.19
	NMTR1	338	33	2	1.10

Table 3: Comparative rankings for SBMIN, DXZ and NMTR1 with band preconditioning on 373 problems

These rankings indicate that, on a problem by problem basis, both the two non-monotone methods outperform SBMIN for all three criteria, NMTR1 being the clear winner. The behaviour of NMTR1 is coherent with the average behaviour reported in Table 2, but the good performance of DXZ indicates that the poorer average results for this algorithm are due to its very bad runs on only a limited number of problems.

However, these numbers as well as additional experiments also reported in Toint (1994b) indicate that the approach taken in NMTR1 is preferable, in our experience, to that proposed by Deng, Xiao and Zhou. As a consequence, we will not discuss Algorithm DXZ any further, but rather concentrate on the further exploration of the potential of NMTR1 as compared to SBMIN.

The next step in our experiments is to analyse the sensitivity of Algorithm NMTR1 with respect to the parameter p, which measures "how far in the past" one is ready to return to define a sufficient reduction. We therefore ran our band preconditioned Algorithm NMTR1 with a variety of different values of p on our 390 problems and compared the performances obtained for the 379 problems coherently solved by all variants. The results are reported, for values of p that are multiples of five, in Tables 4 and 5.

These results are, in the author's mind, rather surprising, in that they seem to show that a very large value of p, even possibly infinite, gives best performance (see Table 5 in particular).

p	successes	failures	performance on 379 problems		
			iters	CG-its	time
5	381	9	34877	632060	12907
10	382	8	26678	508928	12611
15	383	8	24644	446212	11922
20	381	9	24000	428233	11308
25	382	8	22995	426445	11345
30	381	9	23269	425773	10844
35	382	8	22876	417359	11150
40	382	8	23248	433879	11676
∞	382	8	24628	534729	12761

Table 4: Cumulative statistics for NMTR1 with varying p

If an infinite p is chosen, this is to say that one always considers the complete history of the algorithm and that $m(k) = 0$ in (2.19) and thus $r(k) = 0$ for all k. In other words, all decreases obtained in the course of the algorithm are computed using the starting point as reference! It is the author's opinion that this strategy may put an exaggerated emphasis on this point, a potentially undesirable feature. As a result, it was felt appropriate to investigate adaptive ways to choose p. This is the object of the next section.

4 An adaptive non-monotone trust-region algorithm

The main purpose of the parameter p being to define how far in the past of the current iterate to search for its associated reference iteration, we may view the problem of adaptively determining p as that of adaptively determining the reference iteration itself. Since the experimental results discussed in the previous section tend to show that the same reference iteration should typically be used for many k, it is tempting to only modify the reference iteration if some unwanted behaviour occurs. A further analysis of the worst runs of NMTR1 with $p = \infty$ indicated that difficulties typically arose when the objective function values continued increasing for several successive iterations after reaching a (temporary) minimum. As a consequence, a strategy was designed to prevent such occurences. It is again described as a modification of Step 3 of Algorithm BTR.

In order to describe our proposal, we need to define some more notation. We will denote by $f_{\min}$ the current best value of the objective function over all past iterates, that is, at iteration k,

$$f_{\min} = \min_{i=0,\ldots,k} f(x_i). \tag{4.3}$$

The number of iterations since the value of $f_{\min}$ was actually attained will be denoted by ℓ, and we will decide to reset the reference iteration and its associated function value, denoted f_r, if ℓ exceeds some preset positive integer constant h. When this occurs, we suggest resetting f_r to the maximal value of the objective function observed in the past h iterations, that is since the last

	p	1rst	2nd	3rd	4th	5th	6th	7th	8th	9th	e
iters	5	304	28	12	4	3	8	3	8	8	1.67
	10	310	26	9	5	5	9	4	5	5	1.59
	15	314	18	14	8	6	9	6	2	1	1.52
	20	313	17	18	12	5	6	5	2	0	1.48
	25	318	21	14	10	8	4	2	1	0	1.40
	30	319	22	11	10	8	3	3	2	0	1.40
	35	321	19	13	9	9	1	1	3	2	1.41
	40	324	16	12	8	7	3	5	2	1	1.42
	∞	331	18	10	5	4	2	2	3	3	1.36
CG-its	5	306	27	6	7	4	3	9	7	9	1.70
	10	307	26	11	4	7	7	5	5	6	1.62
	15	318	13	17	6	5	7	7	4	1	1.52
	20	318	9	20	20	7	7	2	3	2	1.51
	25	315	17	14	13	12	3	3	0	1	1.46
	30	318	19	10	14	6	4	3	3	1	1.46
	35	321	15	14	9	6	4	3	4	2	1.47
	40	324	15	13	9	3	6	4	3	1	1.43
	∞	332	12	11	8	1	5	3	2	4	1.40
time	5	331	20	7	7	7	3	2	1	0	1.31
	10	333	17	12	6	5	4	1	0	0	1.28
	15	337	14	13	9	3	1	1	0	0	1.24
	20	339	13	18	6	1	1	0	0	0	1.20
	25	339	19	14	4	1	1	0	0	0	1.18
	30	341	19	10	5	2	1	0	0	0	1.18
	35	342	17	10	6	1	2	0	0	0	1.18
	40	344	14	12	4	2	1	1	0	0	1.18
	∞	353	11	5	3	4	2	0	0	0	1.15

Table 5: Comparative rankings of Algorithm NMTR1 with varying p

overall best value was found. We must thus remember this candidate reference value (denoted f_c) and the associated sum of predicted model decreases on all subsequent successful iterations (denoted σ_c). We finally denote by σ_r the sum of all predicted model decreases on the successful iterations since the reference iteration. It is assumed, in the formal description that follows, that the initializations

$$f_{\min} = f_r = f_c = f(x_0), \quad \ell = 0, \qquad \text{and } \; \sigma_r = \sigma_c = 0 \tag{4.4}$$

are performed at Step 0 of the algorithm.

Step 3 of a non-monotone trust-region algorithm (NMTR2)

Step 3: Acceptance of the trial point.

Step 3a: update the iterate. Compute $f(x_k + s_k)$ and the ratios

$$\rho_{1,k} = \frac{f_r - f(x_k + s_k)}{\sigma_r} \tag{4.5}$$

and

$$\rho_{2,k} = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)} \tag{4.6}$$

and set

$$\rho_k = \max[\rho_{1,k}, \rho_{2,k}]. \tag{4.7}$$

If $\rho_k < \eta_1$, then define $x_{k+1} = x_k$ and go to Step 4. Otherwise define $x_{k+1} = x_k + s_k$ and update σ_c and σ_r according to

$$\sigma_c = \sigma_c + m_k(x_k) - m_k(x_{k+1}) \tag{4.8}$$

and

$$\sigma_r = \sigma_r + m_k(x_k) - m_k(x_{k+1}). \tag{4.9}$$

Step 3b: update the best value found so far.
If $f(x_{k+1}) < f_{\min}$ then set

$$f_c = f_{\min} = f(x_{k+1}), \quad \sigma_c = 0 \text{ and } \ell = 0, \tag{4.10}$$

and go to Step 4. Otherwise, increment ℓ by one.

Step 3c: update the candidate for the next reference value.
If $f(x_{k+1}) > f_c$, set

$$f_c = f(x_{k+1}) \text{ and } \sigma_c = 0. \tag{4.11}$$

Step 3d: possibly reset the reference value.
If $\ell = h$, set

$$f_r = f_c \text{ and } \sigma_r = \sigma_c. \tag{4.12}$$

We note that this new variant, NMTR2, is closer to the "watchdog" technique than NMTR1 in that action is taken only if a preset number of increases in the objective function are observed. We also verifies that it is identical to NMTR1 with infinite p provided an improvement on the best objective function value always occurs within h successive successful iterations. Unlike the slightly simpler NMTR1, NMTR2 does not require any additional vector to be stored, but all relevant quantities can instead be recurred from one iteration to the next.

We again refer the reader to the Appendix for the detailed convergence theory for Algorithm NMTR2.

5 Some numerical experiments with Algorithms BTR, NMTR1 and NMTR2

We also modified SBMIN according to the description suggested in the previous section to obtain an implementation of Algorithm NMTR2 comparable to SBMIN and NMTR1. We then ran

NMTR2 on the same battery of 390 problems which we already used above, using the choice $h = 5$. Tables 6 and 7 give the cumulative statistics for SBMIN, NMTR1 ($p = \infty$) and NMTR2, where all three methods use the five banded preconditioner. A total of 382 problems were coherently solved by all three algorithms.

method	successes	failures	performance on 382 problems		
			iters	CG-its	time
SBMIN	382	8	36526	577318	14376
NMTR1	382	8	24746	537675	12780
NMTR2	382	8	21106	424055	10729

Table 6: Cumulative statistics for SBMIN, NMTR1 ($p = \infty$) and NMTR2 on 390 problems

	method	1st	2nd	3rd	e
iters	SBMIN	277	69	36	1.37
	NMTR1	309	52	21	1.25
	NMTR2	335	33	14	1.16
CG-its	SBMIN	280	67	35	1.36
	NMTR1	303	56	23	1.27
	NMTR2	330	36	16	1.18
time	SBMIN	290	76	16	1.28
	NMTR1	343	32	7	1.12
	NMTR2	365	11	6	1.06

Table 7: Comparative rankings for SBMIN, NMTR1 ($p = \infty$) and NMTR2 on 382 problems

Three conclusions can be drawn from these results.

1. The reliability of NMTR2 matches the excellent results of SBMIN and NMTR1.

2. NMTR2 clearly outperforms NMTR1 by a significant margin and for all three considered criteria.

3. NMTR2 also shows a substantial improvement when compared to SBMIN. The gain is about 30% both for the iteration (and function evaluation) count and for the overall execution time.

It is also natural to examine if NMTR2 is very sensitive to the choice of the parameter h. The results of running NMTR2 on the same 390 problems with varying h are presented in Tables 8 and 9. They show that this sensitivity is relatively low, and also that a choice of $h = 5$ is marginally better than higher values. This is also born out by additional experiments not reported here. We may thus temporarily conclude that NMTR2 appears to be an attractive non-monotone alternative to more classical trust-region techniques.

h	successes	failures	performance on 382 problems		
			iters	CG-its	time
5	382	8	21106	424055	10729
10	382	8	22847	492982	10952
15	382	8	22361	468770	10783

Table 8: Cumulative statistics for NMTR2 on 390 problems with varying h

	h	1st	2nd	3rd	e
iters	5	372	6	4	1.04
	10	367	9	6	1.05
	15	362	15	5	1.07
CG-its	5	370	7	5	1.04
	10	366	11	5	1.05
	15	360	15	7	1.08
time	5	378	3	1	1.01
	10	377	1	4	1.02
	15	375	7	0	1.02

Table 9: Comparative rankings for NMTR2 on 382 problems with varying h

It is however important to verify how this conclusion is affected by our choice of preconditioner. In order to investigate this question, we ran SBMIN and NMTR2 (with $h = 5$) on our 390 problems again, but using two very contrasted preconditioning strategies. We first consider using the identity matrix (i.e. no preconditioning at all). In order to slightly refine our conclusions, we distinguish from now on the results between the three subsets of our 390 problems: small (226 cases), large (101 cases) and constrained (63 cases). Moreover, we concentrate on the comparison between SBMIN and NMTR2.

We first examine the relative reliability of SBMIN and NMTR2 without preconditioner. As indicated in Table 10, both methods behave identically on our test sets.

	small probs		large probs		const. probs	
method	succ	fail	succ	fail	succ	fail
SBMIN	223	3	99	2	54	9
NMTR2	223	3	99	2	54	9

Table 10: Comparative reliability of SBMIN and NMTR2 without preconditioner

We now turn to the cumulative statistics shown in Table 11. In this table and the following ones, the number of problems quoted is that of problems that were coherently solved and hence for which efficiency is comparable. We see in Table 11 that NMTR2 again outperforms SBMIN on average. This conclusion is most apparent for small and large unconstrained problems, but less marked for the constrained test cases. One should however remember here that the comparison is not as direct for constrained problems as it is for unconstrained ones.

	222 small problems			97 large problems			53 constrained problems		
method	iters	CG-its	time	iters	CG-its	time	iters	CG-its	time
SBMIN	11307	101940	1001	8217	304664	11023	23093	3330218	19870
NMTR2	6223	58520	512	5168	140272	7167	17323	3212791	17495

Table 11: Cumulative statistics for SBMIN and NMTR2 without preconditioner

Examining now the comparative rankings for our unpreconditioned methods, we conclude from Table 12 that NMTR2 is the best choice on a large proportion of test problems. In comparing these results with those of Table 11, we notice that the important average gains obtained for the second problem subset are achieved on a relatively small number of cases (the gain of 3856 seconds is only obtained on 16 large problems). This indicates that NMTR2 appears to be especially interesting in cases that are difficult for SBMIN.

	222 small problems			97 large problems			53 const. problems		
	SBMIN wins	tie	NMTR2 wins	SBMIN wins	tie	NMTR2 wins	SBMIN wins	tie	NMTR2 wins
Iterations	9	163	50	12	68	17	2	31	20
CG iterations	9	163	50	11	98	18	4	31	18
Time	2	173	47	10	71	16	1	34	18

Table 12: Comparative rankings for SBMIN and NMTR2 without preconditioner on 222 small, 97 large and 53 constrained problems coherently solved

For the sake of completeness, we now return to the five-banded preconditioner, and detail the comparative performance of SBMIN and NMTR2 for the three problem subsets. The reliability results are presented in Table 13, the cumulative statistics in Table 14 and the relative rankings in Table 15. Of course, the conclusions of these results support our previous comments. One observes however the excellent performance of NMTR2 on small problems, where it is never outperformed by SBMIN.

	small probs		large probs		const. probs	
method	succ	fail	succ	fail	succ	fail
SBMIN	220	6	99	2	63	0
NMTR2	220	6	99	2	63	0

Table 13: Comparative reliability of SBMIN and NMTR2 with band preconditioner

	220 small problems			99 large problems			63 constrained problems		
method	iters	CG-its	time	iters	CG-its	time	iters	CG-its	time
SBMIN	12294	51147	1686	6898	52688	8547	17334	473483	4142
NMTR2	5894	23550	636	4623	32785	6706	10589	367720	3386

Table 14: Cumulative statistics for SBMIN and NMTR2 with band preconditioner

	220 small problems			99 large problems			63 const. problems		
	SBMIN wins	tie	NMTR2 wins	SBMIN wins	tie	NMTR2 wins	SBMIN wins	tie	NMTR2 wins
Iterations	14	154	53	12	67	20	10	25	28
CG iterations	17	151	52	11	66	22	13	25	25
Time	0	179	41	8	70	21	3	31	29

Table 15: Comparative rankings for SBMIN and NMTR2 with band preconditioner

We finally consider a substantially better preconditioner. As the band preconditioner, it is available as an option in the LANCELOT package. It uses a modified sparse factorization of the full Hessian, the modification being made to ensure positive definiteness, according to the technique proposed by Schnabel and Eskow (1991).

We first observe, in Table 16, the higher failure rate on large problems for both methods, as compared to that obtained for the simpler preconditioners. It is partly but not completely explained by failure to accomodate the memory requested for the sparse factorization (2 cases). However, the reliability of NMTR2 is again identical to that of SBMIN.

	small probs		large probs		const. probs	
method	succ	fail	succ	fail	succ	fail
SBMIN	222	4	93	8	61	2
NMTR2	222	4	93	8	61	2

Table 16: Comparative reliability of SBMIN and NMTR2 with full factorization preconditioner

If we finally turn to the comparison of efficiency between NMTR2 and SBMIN with full factorization preconditioner (as shown on average in Table 17 and problem by problem in Table 18), we see a much more blurred picture. Although NMTR2 still wins on average on small and constrained problems, the margin is now much smaller than before. Furthermore, NMTR2 now looses on average cpu-time for the large unconstrained problems (the most significant losses being admittedly concentrated in a few problems: FMINSURF, MSQRTALS, MSQRTBLS). More significantly, NMTR2 losses more often that wins for all criteria on this problem subset. This seems to indicate that, overall, the interest of using a non-monotone method may decrease with the quality of the preconditioner, a conclusion already reached in Toint (1994a) in the context of linesearch algorithms.

	221 small problems			93 large problems			60 constrained problems		
method	iters	CG-its	time	iters	CG-its	time	iters	CG-its	time
SBMIN	18547	242395	13857	5077	9972	15522	12202	105125	3755
NMTR2	8562	16871	13669	4499	8842	21951	10368	102497	3610

Table 17: Cumulative statistics for SBMIN and NMTR2 with full factorization preconditioner

	221 small problems			93 large problems			60 const. problems		
	SBMIN wins	tie	NMTR2 wins	SBMIN wins	tie	NMTR2 wins	SBMIN wins	tie	NMTR2 wins
Iterations	20	164	47	15	72	6	8	28	24
CG iterations	20	162	49	15	71	7	11	28	21
Time	5	172	44	15	72	6	4	33	23

Table 18: Comparative rankings for SBMIN and NMTR2 with full factorization preconditioner

6 Conclusions

We have introduced two new non-monotone trust-region algorithms for nonlinear optimization. The proposed methods are globally convergent and numerically compare favourably with SBMIN, a component of the LANCELOT package, and with an earlier proposal by Deng, Xiao and Zhou. The results obtained indicate that substantial computational benefits can often be obtained by abandoning, at least to some extent, the traditional monotonicity constraint on successive objective function's values.

A few more specific conclusions may also be drawn from all the statistics presented in the paper

1. Generally speaking, the non-monotone trust-region algorithm appears to be very competitive with SBMIN. It seems to offer a comparable reliability and is often more efficient.

2. The gain in using non-monotone technique seemingly decreases when the quality of the preconditioner increases: clearly better for the unpreconditioned and band-preconditioned cases, Algorithm NMTR2 remains comparable with SBMIN when full Hessian preconditioning is used, but the relative advantage is much less obvious.

3. The absolute differences between the two methods are more marked for large problems, because of the more intensive computations involved.

4. As expected, preconditioning does not bring efficiency gains when the dimension of the problem is small. It indeed introduces computational overheads but does not cause a

substantial reduction in the number of CG iterations, mostly because only a few are needed even without preconditioning.

5. The very nonlinear character of some problems makes it extremely difficult to fully understand the impact of non-monotone techniques, if not impossible with the analysis tools presently at our disposal. We may not, therefore, exclude good or bad "luck" when the solution of a single particular problem is envisaged.

Although promising in its current state of development, research on this topic is still very much in infancy. In particular, global analysis tools for optimization problems are necessary for a real understanding of non-monotone trust-region techniques, including their sensitivity to algorithms parameters. Such tools are unfortunately lacking today, making further progress in this direction difficult.

Aside from the intellectual challenge of deeper understanding, it now remains to see whether the proposed approach will prove computationally useful in the longer term, a question which will only be answered by continued experience. The trends observed in our preliminary experience are however encouraging. We thus feel entitled to conclude that non-monotone trust-region methods merit further attention of the research and development community.

7 Acknowledgements

The author is indebted to F. Facchinei, L. Grippo and S. Lucidi for very open discussions and for bringing the material of Xiao and Zhou (1992) and Deng *et al.* (1993) to his attention after much of the work on Algorithm NMTR1 was already completed. Thanks are also due to Y. Xiao for her interest in this work and to A. Sartenaer and K. Kanzow who both contributed to improve the present manuscript.

References

[Bongartz *et al.*, 1994] I. Bongartz, A. R. Conn, Nick Gould, and Ph. L. Toint. CUTE: Constrained and Unconstrained Testing Environment. *ACM Transactions on Mathematical Software*, (to appear), 1994.

[Bonnans *et al.*, 1992] J. F. Bonnans, E. Panier, A. Tits, and J. L. Zhou. Avoiding the Maratos effect by means of a nonmonotone linesearch II. inequality constrained problems - feasible iterates. *SIAM Journal on Numerical Analysis*, 29:1187–1202, 1992.

[Burke and Moré, 1988] J. V. Burke and J. J. Moré. On the identification of active constraints. *SIAM Journal on Numerical Analysis*, 25(5):1197–1211, 1988.

[Burke *et al.*, 1990] J. V. Burke, J. J. Moré, and G. Toraldo. Convergence properties of trust region methods for linear and convex constraints. *Mathematical Programming, Series A*, 47(3):305–336, 1990.

[Calamai and Moré, 1987] P. H. Calamai and J. J. Moré. Projected gradient methods for linearly constrained problems. *Mathematical Programming*, 39:93–116, 1987.

[Chamberlain *et al.*, 1982] R. M. Chamberlain, C. Lemaréchal, H. C. Pedersen, and M. J. D. Powell. The watchdog technique for forcing convergence in algorithms for constrained optimization. *Mathematical Programming Studies*, 16:1–17, 1982.

[Conn *et al.*, 1988] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. Global convergence of a class of trust region algorithms for optimization with simple bounds. *SIAM Journal on Numerical Analysis*, 25:433–460, 1988. See also same journal 26:764–767, 1989.

[Conn *et al.*, 1992] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. LANCELOT*: a Fortran package for large-scale nonlinear optimization (Release A).* Number 17 in Springer Series in Computational Mathematics. Springer Verlag, Heidelberg, Berlin, New York, 1992.

[Conn *et al.*, 1993] A. R. Conn, Nick Gould, A. Sartenaer, and Ph. L. Toint. Global convergence of a class of trust region algorithms for optimization using inexact projections on convex constraints. *SIAM Journal on Optimization*, 3(1):164–221, 1993.

[Conn *et al.*, 1994] A. R. Conn, Nick Gould, A. Sartenaer, and Ph. L. Toint. Convergence properties of minimization algorithm for convex constraints using a structured trust region (revised). Technical Report 94/5, Department of Mathematics, FUNDP, Namur, Belgium, 1994.

[Deng *et al.*, 1993] N. Y. Deng, Y. Xiao, and F. J. Zhou. Nonmonotonic trust region algorithms. *Journal of Optimization Theory and Applications*, 76(2):259–285, 1993.

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

[Facchinei and Lucidi, 1993] F. Facchinei and S. Lucidi. Nonmonotone bundle-type scheme for convex nonsmooth minimization. *Journal of Optimization Theory and Applications*, 76(2):241–257, 1993.

[Fletcher, 1987] R. Fletcher. *Practical Methods of Optimization.* J. Wiley and Sons, Chichester, second edition, 1987.

[Grippo *et al.*, 1986] L. Grippo, F. Lampariello, and S. Lucidi. A nonmonotone line search technique for Newton's method. *SIAM Journal on Numerical Analysis*, 23(4):707–716, 1986.

[Grippo *et al.*, 1989] L. Grippo, F. Lampariello, and S. Lucidi. A truncated Newton method with nonmonotone line search for unconstrained optimization. *Journal of Optimization Theory and Applications*, 60(3):401–419, 1989.

[Grippo *et al.*, 1991] L. Grippo, F. Lampariello, and S. Lucidi. A class of nonmonotone stabilization methods in unconstrained optimization. *Numerische Mathematik*, 59:779–805, 1991.

[Moré, 1983] J. J. Moré. Recent developments in algorithms and software for trust region methods. In A. Bachem, M. Grötschel, and B. Korte, editors, *Mathematical Programming: The State of the Art*, pages 258–287, Berlin, 1983. Springer Verlag.

[Moré, 1988] J. J. Moré. Trust regions and projected gradients. In M. Iri and K. Yajima, editors, *System Modelling and Optimization*, volume 113, pages 1–13, Berlin, 1988. Springer Verlag. Lecture Notes in Control and Information Sciences.

[Panier and Tits, 1991] E. Panier and A. Tits. Avoiding the Maratos effect by means of a nonmonotone linesearch I. general constrained problems. *SIAM Journal on Numerical Analysis*, 28:1183–1195, 1991.

[Schnabel and Eskow, 1991] R. B. Schnabel and E. Eskow. A new modified Cholesky factorization. *SIAM Journal on Scientific and Statistical Computing*, 11:1136–1158, 1991.

[Steihaug, 1983] T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM Journal on Numerical Analysis*, 20(3):626–637, 1983.

[Toint, 1981] Ph. L. Toint. Towards an efficient sparsity exploiting Newton method for minimization. In I. S. Duff, editor, *Sparse Matrices and Their Uses*, pages 57–88, London, 1981. Academic Press.

[Toint, 1988] Ph. L. Toint. Global convergence of a class of trust region methods for nonconvex minimization in Hilbert space. *IMA Journal of Numerical Analysis*, 8:231–252, 1988.

[Toint, 1994a] Ph. L. Toint. An assessment of non-monotone linesearch techniques for unconstrained optimization. Technical Report 94/14, Department of Mathematics, FUNDP, Namur, Belgium, 1994.

[Toint, 1994b] Ph. L. Toint. A non-monotone trust-region algorithm for nonlinear optimization subject to convex constraints: the complete numerical results. Technical Report 94/26, Department of Mathematics, FUNDP, Namur, Belgium, 1994.

[Xiao and Zhou, 1992] Y. Xiao and F. J. Zhou. Nonmonotone trust region methods with curvilinear path in unconstrained optimization. *Computing*, 48:303–317, 1992.

Appendix: Global convergence theory for Algorithms NMTR1 and NMTR2

This appendix is devoted to the proof of global convergence for Algorithms NMTR1 and NMTR2. More precisely, we wish to prove that, under the assumptions stated in the previous sections, all limit points of the sequence $\{x_k\}$ generated by either algorithm are critical for problem (1.1), irrespective of the position of the starting vector x_0 and initial trust region radius Δ_0. The proof of this property is an adaptation of the existing convergence theory for Algorithm BTR (see Toint (1988), for instance) to the more complex case where monotonicity is abandoned. We focus here on the parts of the theory that have to be revised, using the simplifying assumptions of Section 2 to avoid lenghty developments in other parts. We complete this set of assumptions by assuming that X is bounded, and, without loss of generality, that ω is chosen large enough to ensure that

$$\max_{x \in X} \|\nabla^2 f(x)\| \leq \omega. \tag{A.1}$$

The possible extensions of this simple framework are outlined at the end of the appendix.

From a theoretical stand point, NMTR1 and NMTR2 only differ in the way in which, at iteration k, the reference iteration $r(k)$ (and the associated reference objective function value $f(x_k)$) are chosen.

1. In NMTR1, $r(k)$ is chosen as the index of the reference iteration, according to (2.24).

2. In NMTR2, $r(k)$ is not explicitly set by the algorithm. However, it may be implicitly defined, at iteration k by the relation $f_r = f(x_{r(k)})$. At variance with NMTR1, there is no bound on $k - r(k)$: indeed, if descent is obtained at each successful iteration, $r(k) = 0$ for all k.

Despite these differences, the notion of a reference iteration unifies the theoretical framework in which both NMTR1 and NMTR2 may be considered.

In order to build this framework, we first notice that the mechanism of both algorithms, (2.5) and the definition of $r(k)$ allow to state the following simple but crucial property.

Lemma 1 *For each $k \in \mathcal{S}$,*

$$f(x_{r(k)}) - f(x_{k+1}) \geq \eta_1 \kappa \sum_{\substack{j=r(k) \\ j \in \mathcal{S}}}^{k} \alpha_j \min\left[\frac{\alpha_j}{b_j}, \Delta_j\right]. \tag{A.2}$$

We then show that an iteration must be successful if the current iterate is not critical and the trust region radius small enough.

Lemma 2 *Assume that the assumptions made in Section 2 hold. Assume furthermore that $\alpha_k > 0$ and that*

$$\Delta_k \leq \frac{\kappa \alpha_k (1 - \eta_2)}{2\omega}. \tag{A.3}$$

Then iteration k is successful and

$$\Delta_{k+1} \geq \Delta_k. \tag{A.4}$$

Proof. Observe first that (2.8) and the inequalities $\kappa < 1$ and $\omega \geq 1$ imply that

$$\frac{\kappa(1 - \eta_2)}{2\omega} \leq 1. \tag{A.5}$$

Thus condition (2.5), (A.3) and (2.7) imply that

$$m_k(x_k) - m_k(x_k + s_k) \geq \kappa \alpha_k \min\left[\frac{\alpha_k}{b_k}, \Delta_k\right] = \kappa \alpha_k \Delta_k. \tag{A.6}$$

On the other hand, the mean-value theorem, (A.1), (2.7) and (2.3) yield that

$$\begin{aligned} |f(x_k+s_k) - m_k(x_k+s_k)| &\leq |\langle s_k, \nabla^2 f(\bar{x}_k) s_k\rangle - \langle s_k, \nabla^2 m_k(\tilde{x}_k) s_k\rangle| \\ &\leq 2\omega \|s_k\|^2 \\ &\leq 2\omega \Delta_k^2, \end{aligned} \tag{A.7}$$

where $\bar{x}_k$ and $\tilde{x}_k$ belong to the segment $[x_k, x_k+s_k] \subseteq X$. Hence, combining (A.6), (A.7) and (A.3), we obtain that

$$|\rho_{2,k} - 1| = \left| \frac{f(x_k+s_k) - m_k(x_k+s_k)}{m_k(x_k) - m_k(x_k+s_k)} \right| \leq \frac{2\omega}{\kappa \alpha_k} \Delta_k \leq 1 - \eta_2. \tag{A.8}$$

As a consequence $\rho_k \geq \rho_{2,k} \geq \eta_2 \geq \eta_1$ and the iteration is successful. Furthermore, (2.10) ensures that (A.4) holds. □

As a consequence of this property, we may now prove that the radius cannot become too small as long as a critical point is not approached.

Lemma 3 *Assume that the assumptions made in Section 2 hold. Assume furthermore that there exists a constant $\epsilon > 0$ such that $\alpha_k \geq \epsilon$ for all k. Then there is a constant $\nu > 0$ such that*

$$\Delta_k \geq \nu \tag{A.9}$$

for all k.

Proof. Assume that k is the first k such that

$$\Delta_{k+1} \leq \frac{\kappa \epsilon \gamma_1 (1-\eta_2)}{2\omega}. \tag{A.10}$$

Then we have from (2.10) that

$$\Delta_k \leq \frac{\kappa \epsilon (1-\eta_2)}{2\omega} \tag{A.11}$$

and our assumption on α_k implies that (A.3) holds and thus that (A.4) is satisfied. But this contradicts the fact that k is the first iteration such that (A.10) holds, and our initial assumption is therefore impossible. This yields the desired conclusion with $\nu = \kappa\epsilon\gamma_1(1-\eta_2)/2\omega$. □

The two above results are sufficient to analyze the criticality of the unique limit point of the sequence of iterates when there are only finitely many successful iterations.

Theorem 4 *Assume that the assumptions made in Section 2 hold. Assume furthermore that there are only finitely many successful iterations. Then $x_k = x_*$ for all sufficiently large k and x_* is critical.*

Proof. The mechanism of the algorithm ensures that $x_* = x_{k_0+1} = x_{k_0+j}$ for all $j > 0$, where k_0 is the index of the last successful iterate. Moreover, since all iterations are unsuccessful for sufficiently large k, (2.8) and (2.10) imply that the sequence $\{\Delta_k\}$ converges to zero. If $\alpha_{k_0+1} > 0$, Lemma 2 then implies that there exists a successful iteration of index larger than k_0, which is impossible. Hence $\alpha_{k_0+1} = 0$ and x_* is critical. □

Having proved the desired convergence property for the case where $\mathcal{S}$ is finite, we now restrict our attention to the case where there are infinitely many successful iterations. In this context, we start by proving that at least one accumulation point of the sequence of iterates (when it is infinite) must be critical.

Theorem 5 *Assume that the assumptions made in Section 2 hold. Then one has that*

$$\liminf_{k\to\infty} \alpha_k = 0. \tag{A.12}$$

Proof. Assume, for the purpose of a contradiction, that, for all k,

$$\alpha_k \geq \epsilon \tag{A.13}$$

for some $\epsilon > 0$ and consider a successful iteration of index k. Condition (A.2), (2.7), (A.13) and (A.9) then give that

$$f(x_{r(k)}) - f(x_{k+1}) \geq \ell_k \eta_1 \kappa \epsilon \min\left[\frac{\epsilon}{\omega}, \nu\right] > 0, \tag{A.14}$$

where ℓ_k is the number of successful iterations between iterations $r(k)$ and k ($r(k)$ and k included). Since the objective function is bounded below on X, the left-hand side of this inequality is bounded by a quantity independent of k and ℓ_k must therefore satisfy

$$\ell_k \leq \bar{\ell} \tag{A.15}$$

for some $\bar{\ell}$ independent of k. But (A.15) and the existence of infinitely many successful iterations imply that the sequence $\{r(k)\}$ is infinite and the associated sequence $\{f(x_{r(k)})\}$ is strictly decreasing (because of (A.14)). As a consequence and since the objective is bounded below on X, this sequence is convergent. Hence, the left-hand-side and thus the right-hand side of (A.14) both tends to zero, which is impossible because $\ell_k \geq 1$ by definition. Thus (A.13) cannot hold for all k and (A.12) follows. □

This first result corresponds to the usual first step in the convergence analysis of a trust-region algorithm. We now prove that *all* limit points of the sequence of iterates are critical.

Theorem 6 *Assume that the assumptions made in Section 2 hold. Then one has that*

$$\lim_{k\to\infty} \alpha_k = 0. \tag{A.16}$$

Proof. Consider first any such iteration of index $k \in \mathcal{S}$. We see that this iteration has an associated reference iteration $r(k)$ whose iterate $x_{r(k)}$ has been obtained by a previous successful iteration, which, in turn, has an associated reference iteration, ..., up to the point where x_0 is reached by this backwards referencing process. Hence, we may construct, for each $k \in \mathcal{S}$, a sequence of successful iterates indexed by $\{r_1, r_2, \ldots, r_q\}$ such that

$$x_0 = x_{r(r_1)} \;\text{ and }\; x_{r_{j-1}+1} = x_{r(r_j)}, \;\text{ for }\; j = 2, \ldots, q, \;\text{ and }\; x_{r_q} = x_{r(k)}. \tag{A.17}$$

Note that

$$f(x_0) - f(x_{k+1}) = f(x_0) - f(x_{r_1+1}) + \sum_{j=2}^{q}[f(x_{r_{j-1}+1}) - f(x_{r_j+1})] + f(x_{r(k)}) - f(x_{k+1}) \tag{A.18}$$

and (A.2) implies that each term in the right-hand side of this equation is positive, but also, together with (2.7), that

$$f(x_0) - f(x_{r_1+1}) \geq \kappa\eta_1 \sum_{\substack{t=0 \\ t\in\mathcal{S}}}^{r_1} \alpha_t \min\left[\frac{\alpha_t}{\omega}, \Delta_t\right], \tag{A.19}$$

$$f(x_{r_{j-1}+1}) - f(x_{r_j+1}) \geq \kappa\eta_1 \sum_{\substack{t=r_{j-1}+1 \\ t\in\mathcal{S}}}^{r_j} \alpha_t \min\left[\frac{\alpha_t}{\omega}, \Delta_t\right], \tag{A.20}$$

for $j = 2, \ldots, q$, and that

$$f(x_{r(k)}) - f(x_{k+1}) \geq \kappa\eta_1 \sum_{\substack{t=r(k) \\ t\in\mathcal{S}}}^{k} \alpha_t \min\left[\frac{\alpha_t}{\omega}, \Delta_t\right]. \tag{A.21}$$

Summing (A.19) – (A.21), we obtain that

$$f(x_0) - f(x_{k+1}) \geq \kappa\eta_1 \sum_{\substack{t=0 \\ t\in\mathcal{S}}}^{k} \alpha_t \min\left[\frac{\alpha_t}{\omega}, \Delta_t\right] \tag{A.22}$$

for each $k \in \mathcal{S}$.

Assume now, for the purpose of a contradiction, that there is a subsequence of successful iterates, indexed by $\mathcal{N}$, such that

$$\alpha_k \geq 2\epsilon > 0 \tag{A.23}$$

for some $\epsilon > 0$ and for all $k \in \mathcal{N}$, and define $\mathcal{K}$ to be the index set of all successful iterations such that $\alpha_k \geq \epsilon$. Consider a $k \in \mathcal{K}$. Applying (A.22), the inclusion $\mathcal{K} \subseteq \mathcal{S}$ and the definition of $\mathcal{K}$, we obtain that

$$\begin{aligned} f(x_0) - f(x_{k+1}) &\geq \kappa\eta_1 \sum_{\substack{t=0 \\ t\in\mathcal{S}}}^{k} \alpha_t \min\left[\frac{\alpha_t}{\omega}, \Delta_t\right] \\ &\geq \kappa\eta_1 \sum_{\substack{t=0 \\ t\in\mathcal{K}}}^{k} \alpha_t \min\left[\frac{\alpha_t}{\omega}, \Delta_t\right] \\ &\geq \kappa\eta_1\epsilon \sum_{\substack{t=0 \\ t\in\mathcal{K}}}^{k} \min\left[\frac{\epsilon}{\omega}, \Delta_t\right]. \end{aligned} \tag{A.24}$$

But the left-hand side of this inequality is bounded because $f(x)$ is bounded below on X, and we see thus that

$$\min\left[\frac{\epsilon}{\omega}, \Delta_t\right] = \frac{\epsilon}{\omega} \tag{A.25}$$

can only be true a finite number of times. Thus there must exist a $k_0 \in \mathcal{K}$ such that

$$\min\left[\frac{\epsilon}{\omega}, \Delta_t\right] = \Delta_t \tag{A.26}$$

for all $t \in \mathcal{K}$, $t \geq k_0$. Therefore, (A.24) implies that

$$\sum_{\substack{t=k_0 \\ t\in\mathcal{K}}}^{k} \Delta_t \leq \frac{1}{\kappa\eta_1\epsilon}\left(f(x_0) - \min_{x\in X} f(x)\right) \tag{A.27}$$

for all $k \in \mathcal{K}$, $k \geq k_0$. As a consequence,

$$\sum_{\substack{t=k_0 \\ t \in \mathcal{K}}}^{\infty} \Delta_t < \infty. \tag{A.28}$$

Theorem 5 and the definition of $\mathcal{K}$ now imply that, for any $k \in \mathcal{N} \subseteq \mathcal{K}$, there must be a smallest $\ell(k) > k$ with $\ell(k) \in \mathcal{S} \setminus \mathcal{K}$. Furthermore, since

$$\begin{aligned} \|x_{\ell(k)} - x_k\| &\leq \sum_{\substack{j=k \\ j \in \mathcal{S}}}^{\ell(k)-1} \|x_j - x_{j+1}\| \\ &\leq \sum_{\substack{j=k \\ j \in \mathcal{S}}}^{\ell(k)-1} \Delta_j \\ &= \sum_{\substack{j=k \\ j \in \mathcal{K}}}^{\ell(k)-1} \Delta_j, \end{aligned} \tag{A.29}$$

we obtain from (A.28) that $\|x_{\ell(k)} - x_k\|$ tends to zero as k tends to infinity. By continuity of the criticality measure, we thus deduce that $\alpha_k - \alpha_{\ell(k)}$ also tends to zero with k. However this is impossible because the definitions of $\mathcal{N}$, k and $\ell(k)$ imply that $\alpha_k - \alpha_{\ell(k)} \geq \epsilon$. Hence, no subsequence satisfying (A.23) can exist and the theorem is proved. □

We have thus shown that the modification of Step 3 defining Algorithms NMTR1 and NMTR2 are consistent with the general framework of trust-region methods, in that convergence properties standard for this class are preserved.

We conclude this discussion by indicating that the theory developed in this section can be generalized in various ways. We have consciously used a simple framework for our analysis, but it can be extended to cover, for example, the case where the feasible set is unbounded (the assumption that X is bounded is then replaced by the assumption that the level set corresponding to the starting point is bounded or, more generally, that the iterates remain in some bounded region). This is important because it indicates that Algorithms NMTR1 and NMTR2 may be applied to unconstrained problems with the same theoretical convergence properties remaining valid. Other possible extensions include scaled iteration dependent norms for the definition of the trust region in (2.3), the use of approximate gradient vectors, less restrictive assumptions on the growth of the parameters b_k in (2.6) and the use of problem structure. We refer the reader to Conn *et al.* (1994) for a discussion of these issues in the monotone context.