Operations Research Letters 46 (2018) 7–12

Contents lists available at ScienceDirect

Operations Research Letters

journal homepage: www.elsevier.com/locate/orl

An adaptive truncation criterion, for linesearch-based truncated Newton methods in large scale nonconvex optimization

Andrea Caliciotti [a], Giovanni Fasano [b], Stephen G. Nash [c], Massimo Roma [a,*]

[a] Dipartimento di Ingegneria Informatica, Automatica e Gestionale "A. Ruberti" SAPIENZA, Università di Roma, via Ariosto, 25 –00185 Roma, Italy
[b] Department of Management, University Ca' Foscari of Venice, S. Giobbe, Cannaregio 873 –30121 Venice, Italy
[c] Systems Engineering & Operations Research Department, George Mason University, 4400 University Drive Fairfax –VA 22030, USA

ARTICLE INFO

Article history:
Received 6 June 2016
Received in revised form 18 October 2017
Accepted 18 October 2017
Available online 28 October 2017

Keywords:
Large scale nonconvex optimization
Linesearch-based truncated Newton methods
Krylov subspace methods
Adaptive truncation criterion

ABSTRACT

Starting from the paper by Nash and Sofer (1990), we propose a heuristic adaptive truncation criterion for the inner iterations within linesearch-based truncated Newton methods. Our aim is to possibly avoid "over-solving" of the Newton equation, based on a comparison between the predicted reduction of the objective function and the actual reduction obtained. A numerical experience on unconstrained optimization problems highlights a satisfactory effectiveness and robustness of the adaptive criterion proposed, when a residual-based truncation criterion is selected.

1. Introduction

In this paper we consider the unconstrained optimization problem

$$\min f(x), \tag{1}$$

where $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ is a real valued function possibly nonconvex, and n is large. We consider the standard assumptions that f is twice continuously differentiable, and that for a given $x_0 \in \mathbb{R}^n$ the level set $\Omega_0 = \{x \in \mathbb{R}^n \mid f(x) \leq f(x_0)\}$ is compact.

Truncated Newton methods are widely used for solving such problems. They are also called Newton–Krylov methods since a Krylov subspace method is usually employed, for approximately solving the Newton equation at each iteration. A general description of the truncated Newton methods can be found in the survey paper [17]. Now we briefly recall the main features of this class of methods. In the sequel, we denote by $f_k = f(x_k)$, $g_k = \nabla f(x_k)$ and $H_k = \nabla^2 f(x_k)$ respectively the function, the gradient and the Hessian matrix of the function f at the point x_k. Moreover, we do not assume any sparsity pattern for H_k.

It is well known that, given an initial guess x_0 of a local minimizer of problem (1), a truncated Newton method is based on two nested loops: the *outer iterations* which represent the actual steps of the method, where the current estimate of the solution is updated; the *inner iterations* which carry out an iterative algorithm for computing, at each outer iteration k, a search direction d_k by approximately solving the Newton equation

$$H_k d = -g_k. \tag{2}$$

The iterative algorithm used for solving (2) is actually "truncated", i.e. terminated before the exact solution is obtained. This strategy is based on the fact that, since the benefits of using a Newton direction are local, i.e. in the neighborhood of a stationary point, an accurate solution of (2) may be unjustified when x_k is far from a local optimizer. As matter of fact, in this case, a much simpler search direction can often perform comparably well. Instead, more accuracy is required when the iterates approach a local minimizer. A good trade-off between the accuracy in solving the Newton equation (2) and the computational effort employed per outer iteration is a key point, for the overall efficiency of a truncated Newton method. Indeed, there might be significant advantages to terminating the inner iterations early, when we are still far from a solution and the problem has significant nonlinearities. On the contrary, when close to a solution, there may be disadvantages to early terminating, inasmuch as the corresponding search direction d_k might be poor.

Since the early papers [4,5] where truncated Newton methods were introduced, the importance of an efficient truncation criterion for the inner iterations was pointed out. The stopping rule proposed therein is based on controlling the magnitude of the residual. Under suitable assumptions, this allows to guarantee local convergence and a good convergence rate. Another truncation rule

* Corresponding author.
E-mail addresses: caliciotti@dis.uniroma1.it (A. Caliciotti), fasano@unive.it (G. Fasano), snash@gmu.edu (S.G. Nash), roma@dis.uniroma1.it (M. Roma).

https://doi.org/10.1016/j.orl.2017.10.014
0167-6377/

has been proposed in [19]. It is based on a comparison between the reduction of the quadratic model of the objective function, at the current iteration, and the average reduction per iteration of the model.

The proper choice of a suitable truncation criterion, along with the necessity to handle the indefinite case (see also [8]) and the choice of an effective preconditioning strategy, are three "open questions" within truncated Newton frameworks, listed in the early survey paper [21]. In subsequent years, even if the research activity on these topics was greatly developed, actually no definitive answer has been yet provided.

Another fundamental aspect of a truncated Newton method concerns the global convergence. Indeed, as well known, a globalization strategy must be adopted to this aim, i.e. the method must be embedded within a linesearch or a trust region framework.

On the basis of these remarks, in this paper we propose a simple adaptive truncation criterion for the inner iterations within a linesearch-based truncated Newton method, and analyze its effectiveness in both the unpreconditioned and the preconditioned case. Our aim is to define an additional rule which enables to avoid "over-solving" of the Newton equation (2) in some circumstances. The latter phenomenon occurs whenever unnecessary inner iterations are performed, so that indulging in solving the Newton equation does not produce a better search direction. This possibly yields a reduction of the overall inner iterations, for both convex and nonconvex problems. Our proposal is partially inspired by trust region approach (see e.g. [3]), and is based on a comparison between the reduction of the objective function predicted by the quadratic model, and the actual reduction obtained. In particular, we consider a linesearch–based truncated Newton method where the inner iterations are performed using the Conjugate Gradient (CG) algorithm. A numerical experience was carried on for a selection of large scale (convex and nonconvex) test problems from CUTEst collection [11], showing the satisfactory effectiveness and the robustness of the adaptive rule proposed, when the residual-based criterion is adopted. For the sake of brevity, we only report a few significant summary results. The complete numerical results are detailed in the companion paper [1].

The paper is organized as follows: in Section 2 the two commonest truncation criteria used in the literature are reported and discussed. Section 3 reports some introductory theoretical motivations which are at the basis of our proposal. The novel adaptive rule is introduced in Section 4 and a summary of the numerical testing is reported in Section 5. Finally, Section 6 reports some concluding remarks. As regards the notations, $\|v\|$ denotes the 2-norm of the vector $v \in \mathbb{R}^n$.

2. Common truncation criteria

In order to briefly recall the truncation criteria commonly used in the literature, we denote by d_k an approximate solution of (2), and by $r_k = H_k d_k + g_k$ the corresponding residual.

A natural stopping criterion for the inner iterations is the *residual-based criterion*, proposed in the seminal papers [4] and [5]. Indeed, the authors propose to terminate the inner iterations whenever the residual r_k is sufficiently small, namely

$$\frac{\|r_k\|}{\|g_k\|} \leq \eta_k, \tag{3}$$

for a specified value of η_k. It is well known that criterion (3) is scale invariant and that the choice of the *forcing sequence* $\{\eta_k\}$ is crucial for controlling the convergence rate of the algorithm. A widely used choice proposed in [5] is $\eta_k = \min\{1/k, \|g_k\|^r\}$, with $0 < r \leq 1$. Other forcing sequences have been proposed later in [6] and [7]. The possibility to easily control the rate of convergence of the algorithm, by means of suitable choices of the sequence $\{\eta_k\}$, is a key point for this criterion. On the other hand, some drawbacks deriving from the adoption of this rule are well known. Indeed, at the jth inner iteration of the Krylov subspace method adopted for solving the Newton equation (2), a stationary point of the quadratic model

$$q_k(d) = \frac{1}{2} d^T H_k d + g_k^T d \tag{4}$$

over the Krylov subspace

$$\mathcal{K}_j(H_k, g_k) = span\left\{g_k, H_k g_k, H_k^2 g_k, \ldots, H_k^{j-1} g_k\right\}$$

is sought. In the case of positive definite Hessian H_k, the quadratic model (4) has a global minimizer which exactly solves the Newton equation (2). Of course, this case corresponds to a null residual. Conversely, whenever an approximate solution is sought, monitoring the magnitude of the residual might be, as discussed in [19], misleading. Indeed, the actual decrease of the objective function values can be alternatively predicted by means of the quadratic model decrease; however, the magnitude of the residual r_k and the quadratic model $q_k(d_k)$ could be significantly different. Moreover, the rounding error in computing $\|r_k\|$ could be relevant since r_k is usually computed by recurrence. In addition, whenever a CG method is used in the inner iterations and H_k is positive definite, the quadratic model monotonically decreases as the inner iterations progress, while the sequence $\{\|r_k\|\}$ is not monotone (unlike, for instance, using MINRES).

These remarks induced the authors to propose in [19] also a truncation rule based on *the decrease of the quadratic model*, rather than considering only the residual. Namely, the truncation criterion proposed is the following: the inner iterations are terminated if, for a specified value of $\eta_k \in (0, 1)$,

$$\frac{q_k(d_j) - q_k(d_{j-1})}{\frac{q_k(d_j)}{j}} \leq \eta_k, \tag{5}$$

where d_j denotes the approximate solution of (2) at the jth inner iteration. This criterion is then based on the comparison between the reduction of the quadratic model $q_k(d_j) - q_k(d_{j-1})$, and the average reduction per iteration $q_k(d_j)/j$. The criterion (5) is often considered preferable to (3) , since it gains information directly from the values of the quadratic model. Moreover, in [9] it was extended to possibly consider also an indefinite Hessian matrix H_k, providing some theoretical results, too.

However, in the framework of truncated Newton methods, in unconstrained as well as in constrained optimization, some codes currently available on the web and commonly used by the optimizers community still adopt the residual-based truncation criterion (3), both within linesearch-based and trust region-based codes (see e.g. [2,13], [15] page 9, [12,22–24] and URL https://neos-guide.org/content/truncated-newton-methods). This might be due also to the fact that, as well known, the adoption of (3) ensures theoretical superlinear convergence. Conversely, the criterion based on the quadratic model reduction (5), with the suggested value of $\eta_k = 0.5$ (constant), guarantees only the theoretical linear rate of convergence [19], even if it works very efficiently in practice.

On the basis of these remarks, in this paper we focus on the possibility to "enrich" the residual-based criterion (3) by conveying, also in this case, information gained from the behavior of the quadratic model. To this aim, we propose an adaptive rule for deciding the maximum number of inner iterations allowed at each outer iteration. The latter rule combined with the criterion (3) should enhance the overall efficiency of a truncated Newton method, by possibly avoiding the over–solving phenomenon.

3. Motivation for the truncation rule

Both the stopping criteria (3) and (5) in the previous section may not prevent over-solving of the Newton equation. For the

residual-based criterion (3), different forcing sequences have been proposed to stem this phenomenon [7], still guaranteeing a good asymptotic rate of convergence.

As regards the criterion (5), it is well-grounded if the quadratic model is accurate, i.e. if the quadratic model is a good (local) approximation of the objective function. Conversely, if accuracy is poor, a successful strategy has been proposed in [20] in the context of block-truncated Newton methods. The strategy is simple and consists of allowing only one inner iteration if the stepsize determined by the linesearch procedure in the previous outer iteration is different from one. The rationale behind this strategy relies on the fact that a stepsize different from one (i.e. the search direction likely does not resemble the Newton direction) means that the quadratic model is likely inaccurate. In the context of a block method where each inner iteration represents a significant computation, this simple rule was appropriate and effective. When the standard CG method is used as the inner iteration, a more nuanced strategy is desired.

Additional justification for avoiding over-solving is provided by the computational results in [18]. This paper compares the performance of a truncated Newton method and a limited-memory BFGS method, and concludes that the truncated Newton method displays superior performance when the quadratic model is a good approximation to the objective function.

Based on this evidence, we propose an adaptive rule for dynamically setting the maximum number of inner iterations at each outer iteration of a linesearch-based truncated Newton scheme, possibly allowing the Hessian matrix H_k to be indefinite. In particular, inspired by trust region methods [3] (and borrowing their terminology/notation), at each outer iteration k, our idea is to compare the *actual reduction* of the objective function

$$Ared_k = f_k - f(x_k + s_k)$$

with the *predicted reduction*, i.e. the reduction predicted by the quadratic model

$$Pred_k = q_k(0) - q_k(s_k) = -\left[\frac{1}{2} s_k^T H_k s_k + g_k^T s_k\right],$$

where $s_k = \alpha_k d_k$ and α_k is the stepsize computed by the linesearch procedure. Our truncation criterion will be based on the difference between actual and predicted reduction, an estimate of the difference between the quadratic model and the objective function. It is this quantity that was determined in [18] to be significant to the performance of a truncated Newton method.

To provide further insight into this choice, we examine the difference between *Ared* and *Pred*. If they are similar then we will conclude that the quadratic model is a good approximation to the objective function, and that the inner iteration is computing an effective search direction. Our focus is on the difference between the quadratic model and the higher-order terms in the Taylor series approximation to f, as motivated by the comments above.

Let us look at these quantities in more detail. Since $f \in C^2(\mathbb{R}^n)$,

$$Ared_k = f_k - f(x_k + s_k) = -{s_k}^T g_k - \frac{1}{2} {s_k}^T H(x_k + \xi s_k) s_k,$$

where $0 \le \xi \le 1$. For $Pred_k$ we obtain

$$Pred_k = f_k - [f_k + {s_k}^T g_k + \frac{1}{2} {s_k}^T H_k s_k] = -{s_k}^T g_k - \frac{1}{2} {s_k}^T H_k s_k.$$

Combining these, we have

$$Ared_k - Pred_k = \frac{1}{2} {s_k}^T [H_k - H(x_k + \xi s_k)] s_k.$$

Now, observe that if $f(x)$ is a quadratic function then

$$H_k = H(x_k + \xi s_k)$$

yielding

$$Ared_k - Pred_k = 0.$$

On the contrary, if $f(x)$ is not quadratic and the quadratic model is not a good approximation to it, then the difference $Ared_k - Pred_k$ will be large. Similarly, if $\|s_k\|$ is small, this difference will be small, as we can expect when the truncated Newton algorithm converges. Thus, on balance, we can monitor values of $|Ared_k - Pred_k|$ to possibly introduce an adaptive truncation criterion.

Our focus on *Ared* and *Pred* is reminiscent of trust region methods, but our approach is distinct. In a trust region method, if there is disagreement between *Ared* and *Pred* then a bound on the norm of the search direction is reduced. In our case (see Section 4) we will reduce a bound on the number of inner iterations, i.e., a bound on the computational effort. As motivated by [18], if the quadratic model is not a good approximation to the objective function, it is not worthwhile to use a large number of inner iterations to minimize the quadratic model, to compute a search direction.

There is a relationship between the trust region approach and our approach. When the CG method is used in the inner iteration, the estimate of the search direction s_k increases monotonically in norm (provided that a suitable norm is adopted), at each iteration. Hence bounding the norm of s_k will limit the number of inner iterations, and *vice versa*, but the relationship between the two approaches is not precise. Even if the bound on s_k is small a significant number of CG iterations might still result.

This is analogous to the relationship between the residual-based stopping criterion (3) and the quadratic-based stopping criterion (5). There is a theoretical relationship between them [16], but the latter is based directly on limiting the computational effort if it is determined that the inner iteration is not contributing to the progress of the optimization. As shown in [19], the norm of the residual can be a poor predictor of the quality of the search direction. It is our hope that we can reduce the effect of over-solving by focusing directly on the computational effort in the algorithm.

Our adaptive truncation criterion (defined in Section 4) will be based on $|Ared - Pred|$. We will be assessing how well the quadratic model approximates the objective function. This is different than in a traditional trust region method, where it is more common to assess only whether the predicted reduction underestimates the actual reduction.

4. Our adaptive truncation criterion

The analysis of the previous section can be used for defining an Adaptive Truncation Criterion (ATC) within a truncated Newton scheme. The quantity

$$\rho_k = \left| Ared_k - Pred_k \right| \tag{6}$$

is at the basis of our adaptive rule. In particular, we adaptively set the maximum number max_it_{k+1} of inner iterations allowed at the outer iteration $k+1$, on the basis of the value of ρ_k.

The Adaptive Truncation Criterion we propose is detailed in the following scheme:

Adaptive Truncation Criterion (ATC)

Data: $0 < \gamma_1 < \gamma_2$, $0 < \sigma_3 < 1 < \sigma_2 < \sigma_1$, $0 < \theta_2 < \theta_1$ and $\ell \in \mathbb{N}$, $1 \le \ell < n$.

If $\rho_k \le C_k \gamma_1$ then (*very successful step*)
  if $\alpha_k \ge \theta_1$ then set $max\_it_{k+1} = \min\{n\ ,\ \lfloor \sigma_1 max\_it_k \rfloor\}$

else if $\rho_k \le C_k \gamma_2$ then (*successful step*)
  if $\alpha_k \ge \theta_2$ then set $max\_it_{k+1} = \min\{n\ ,\ \lfloor \sigma_2 max\_it_k \rfloor\}$

else (*unsuccessful step*)
  set $max\_it_{k+1} = \max\{\ell\ ,\ \lfloor \sigma_3 max\_it_k \rfloor\}$

The maximum number of inner iterations is increased in case of *successful steps*, otherwise it is decreased (*unsuccessful steps*). Note that in the *successful steps* an additional check on the stepsize α_k is introduced. This is motivated by the fact that when d_k is poor, then $\|s_k\|$ can actually be very small after the linesearch procedure, possibly yielding an unexpected *successful*/*very successful* step. To prevent the latter drawback, following the rationale behind the proposal in [20], we verify that the quality of s_k resembles d_k, by checking the steplength α_k. Whenever the stepsize is too small, then d_k is likely poor and we leave max_it_k unchanged.

The quantity C_k is introduced in order to take into account the magnitude of $Ared_k$ and $Pred_k$, so that the adopted test is well scaled. A detailed discussion about possible choices for C_k is reported in Section 5. The purpose of the threshold value ℓ is to guarantee that a certain number $\ell \ll n$ of inner iterations is anyhow performed. This value plays an important role whenever the information collected during the inner iterations is possibly used to construct preconditioners [9] and [10]. Indeed, in this case a threshold number of inner iterations should prevent the construction of an unreliable preconditioner. Of course, other possible preconditioning strategies might differently affect the choice of parameter ℓ. Observe that, since in ATC we have $\ell \geq 1$, and since the first CG iteration produces a search direction proportional to the negative gradient, the ATC strategy always yields a gradient-related direction, which guarantees global convergence.

We can summarize the importance of the ATC criterion by observing that it complies with the following three issues:

1. it aims at extending the strategy in [20] already mentioned;
2. it attempts to partially exploit second order information on the objective function (including also the indefinite case), by considering in (6) a quadratic model update;
3. considering that we are dealing with large scale problems, it does not require significant additional computational burden and supplementary storage.

In order to clarify the latter key points, we first observe that the strategy in [20] substantially uses information from the linesearch procedure, to infer second order information on the function. Indeed, whenever the stepsize is equal to one, then the search direction is a Newton-like direction, implying that the local second order model is a "qualified" approximation of the objective function. In this regard, ρ_k should be, to some extent, a measure of this "qualification". As regards item 2., note that ρ_k includes information on second order derivatives of f, throughout the computation of the quadratic model. Thus, ρ_k possibly summarizes some second order information on f, too. Finally, as concerns item 3., considering the large scale setting, the computational cost of ATC is definitely negligible.

Note that the early termination of the inner iterations is equivalent to restarting the iterative method used (say the CG method). In this regard, the use of a preconditioner (if any) could be helpful in avoiding a possible deterioration in performance due to this restart. This motivates the fact that in the numerical experiences we also include the use of a preconditioning strategy, combined with ATC. The considerations in the current paragraph deserve a more accurate analysis based on numerical experiences, as reported in the next section and in [1].

We conclude this section by observing that, as already said, the rule we proposed is based on ρ_k. Since in (6) the predicted reduction is based on the quadratic Taylor series approximation, ρ_k can be bounded in terms of the third derivatives of the objective function and the magnitude of the current search direction. As the algorithm converges, ρ_k goes to zero, and our adaptive truncation criterion reverts to a traditional truncated Newton method.

5. Numerical experiences

In this section, we report a brief summary of an extensive numerical testing by considering a standard implementation of a truncated Newton method. The complete results are included in the companion paper [1]. The Conjugate Gradient method is employed in the inner iterations. The novelty consists in the adoption of the adaptive criterion described in the previous section. Thus the maximum number of CG inner iterations allowed per outer iteration (initialized to n, i.e. $max\_it_1 = n$) is adaptively adjusted according to ATC. We tested ATC both within the unpreconditioned and the preconditioned framework proposed in [9]. As regards the stopping criterion for the CG inner iterations, we tested both the criteria recalled in Section 2: the residual-based criterion (3) and the quadratic model reduction-based criterion (5).

We remark that our main goal is to provide an adaptive rule to enhance the residual-based criterion (3). Nevertheless, for completeness we also coupled ATC with criterion (5), though no significant improvement was expected, since (5) already contains second order information.

5.1. Guidelines for the choice of C_k in ATC scheme

The quantity C_k in ATC plays the role of a scaling factor with respect to values of the objective function, which should duly take into account the size of $Ared_k$ and $Pred_k$. In particular, among several possibilities we tested the following two expressions of C_k

$$C_k = \min\{1, |f(x_k)|\} \tag{7}$$

$$C_k = \max\{1, |f(x_k)|\}, \tag{8}$$

whose rationale may be interpreted as follows. The expression (7) takes into account scaling of the function when $f(x_k)$ is relatively small (i.e. $|f(x_k)| \leq 1$). On the other hand, the expression (8) for C_k takes into account scaling when $f(x_k)$ is relatively large. We experimented both the choices on the whole test set and, though apparently the choice (8) might be more intuitive, setting C_k as in (7) yields appreciably better results (at least on the test set considered). Note that with the choice (7), whenever $|f(x_k)|$ is small, we have C_k close to zero. However, in this case we expect that also $Ared_k$ and $Pred_k$ are not relatively large. Therefore, if C_k is close to zero and the test in ATC scheme fails, we expect that $Ared_k$ and $Pred_k$ differ appreciably, hence the step must be considered unsuccessful. In this regard, the choice for considering the step successful or unsuccessful is completely determined by the value of the parameter γ_1 and γ_2. The results concerning the unpreconditioned and preconditioned cases are very similar. On the basis of these results, we adopt $C_k = \min\{1, |f(x_k)|\}$ for our numerical experience reported in the sequel.

5.2. Numerical comparisons among different schemes

We now summarize the main results of the numerical experiences, reported in [1], namely the use of our adaptive truncation criterion ATC. In particular, our aim is to assess the improvement of using (3) + ATC w.r.t. (3) (in short, *ATC-true* vs. *ATC-false*). Moreover, we also consider results obtained by using (5). We first observe that the adoption of ATC affects the results for several test problems, both in the unpreconditioned and preconditioned case. By observing the performance profiles reported in [1] we deduce that the choice *ATC-true* outperforms *ATC-false* both in terms of number of CG inner iterations and CPU time. This confirms the expectation of our proposal, i.e., coupling the rule (3) with ATC, in practice enhances first order information with some second order information, thus improving the overall performance.

Furthermore, a comparison between the rule (3) with *ATC-true* and (5) is also considered, showing that, in practice, (3)+ ATC both retains the appealing theoretical convergence properties of using (3) and, to some extent, performs similarly to (5). This reveals also that possibly the residual-based criterion does not include enough information on the reduction of the quadratic model, and that the joint use with ATC enables the partial recovery of this information.

As a matter of fact, pairing (3) with ATC proves to be successful, while coupling ATC with (5) risks to spoil the information on second order derivatives, thus yielding inefficiency.

5.3. Comparison with trust region approach

Our ATC rule proposed in this paper, to a large extent, draws inspiration from trust region methods, so that it could be significant to directly compare our (linesearch-based) approach versus a trust region approach. We want to assess the behavior of a truncated Newton method based on our ATC strategy, with respect to a standard trust region code. On this purpose we consider the TRON code [13] which represents one of the most commonly used implementations of a trust region truncated Newton method, for large bound-constrained problems (this code is available from Jorge Moré's web page). In this algorithm a descent direction for the trust region subproblem is generated, by means of a preconditioned conjugate gradient method, and the CG iterations are stopped whenever the trust region bound is violated, a negative curvature is encountered or a convergence condition is satisfied. For all the details we refer to [13].

We compare the results obtained by TRON, on the whole test set, versus those obtained by our approach. First, observe that, due to differences among the computational schemes used, the CPU time is likely the most significant indicator to assess the overall computational burden. Hence, it could be misleading to draw any conclusions by comparing, for instance, the number of outer iterations/function evaluations. We run TRON by using all the default parameters of the code and by using both the default stopping criterion and the standard one in [14]. Of course, the default stopping criterion is tighter than the one in [14], so that the use of the stopping criterion in [14] enables early stops for TRON code. As consequence, by using the stopping criterion in [14] in place of the default one, TRON recovers some failures for CPU time limit or number of function evaluations.

On this guideline, in order to make a fair comparison with our proposal, we compare the results obtained by our preconditioned truncated Newton method and by TRON, using the same default stopping criterion for both the algorithms. This comparison is reported in [1], showing that our algorithm outperforms TRON in terms of CPU time, even if in terms of function evaluations and CG inner iterations TRON is more efficient than our proposal. The reason of this is clearly evidenced by observing some runs of TRON. Indeed, on some large scale problems the Incomplete Cholesky Factorization (ICF) used by TRON, when computing the step in the trust region subproblem, is definitely time consuming. This implies that, in solving some difficult large scale problems, after 900 seconds only one or two outer iterations are possibly performed.

As regards the robustness, TRON is much successful (in terms of number of function evaluations and CG inner iterations), while it tends to lack robustness. In order to better clarify the latter issue, the detailed results for all the problems where at least one of the algorithms fails to converge are reported in [1].

On summary, to large extent our proposal compares favorably with respect to TRON.

6. Conclusions

In this paper we addressed the problem of "over-solving" the Newton equation within linesearch-based truncated Newton methods. An adaptive rule for dynamically setting the maximum number of inner iterations, at each outer iteration, is studied. It can be used jointly with any truncation criterion. A significant numerical study has been performed, in order to assess the effectiveness and the robustness of the joint use of the adaptive rule and the two most popular truncation criteria.

Overall, even if a careful tuning of the parameters used in the adaptive rule is certainly still needed, the results obtained definitely agree with those reported in [19] and [20], from which we have drawn inspiration for this work. Finally, we assert that the results obtained seem to indicate that the use of the adaptive rule is promising, particularly in tackling large scale difficult nonconvex problems.

Acknowledgments

The work of G. Fasano is partially supported by the Italian Flagship Project RITMARE, coordinated by the Italian National Research Council (CNR) and funded by the Italian Ministry of Education, within the National Research Program 2012–2016.

References

[1] A. Caliciotti, G. Fasano, S. Nash, M. Roma, Data and performance profiles applying an adaptive truncation criterion, within linesearch-based truncated Newton methods, in large scale nonconvex optimization, Data Brief (2017) (in press).
[2] A.R. Conn, N.I.M. Gould, P.L. Toint, LANCELOT: A Fortran Package for Large-Scale Nonlinear Optimization (Release A), Springer Verlag, Heidelberg, Berlin, 1992.
[3] A.R. Conn, N.I.M. Gould, P.L. Toint, Trust–region methods, MPS–SIAM Series on Optimization, Philadelphia, PA, 2000.
[4] R. Dembo, S. Eisenstat, T. Steihaug, Inexact Newton methods, SIAM J. Numer. Anal. 19 (1982) 400–408.
[5] R. Dembo, T. Steihaug, Truncated–Newton algorithms for large-scale unconstrained optimization, Math. Program. 26 (1983) 190–212.
[6] S. Eisenstat, H. Walker, Globally convergent inexact Newton methods, SIAM J. Optim. 4 (1994) 393–422.
[7] S. Eisenstat, H. Walker, Choosing the forcing term in an inexact Newton method, SIAM J. Sci. Stat. Comput. 17 (1996) 16–32.
[8] G. Fasano, M. Roma, Iterative computation of negative curvature directions in large scale optimization, Comput. Optim. Appl. 38 (2007) 81–104.
[9] G. Fasano, M. Roma, Preconditioning Newton—Krylov methods in nonconvex large scale optimization, Comput. Optim. Appl. 56 (2013) 253–290.
[10] G. Fasano, M. Roma, A novel class of approximate inverse preconditioners for large scale positive definite linear systems in optimization, Comput. Optim. Appl. 65 (2016) 399–429.
[11] N.I.M. Gould, D. Orban, P.L. Toint, CUTEst: A constrained and unconstrained testing environment with safe threads, Comput. Optim. Appl. 60 (2015) 545–557.
[12] A.F. Izmailov, A.I.M. Solodov, A truncated SQP method based on inexact interior–point solutions of subproblems, SIAM J. Optim. 20 (5) (2010) 2584–2613.
[13] C.-J. Lin, J. Moré, Newton's method for large bound–constrained optimization problems, SIAM J. Optim. 9 (1999) 1100–1127.
[14] J. Morales, J. Nocedal, Automatic preconditioning by limited memory quasi–Newton updating, SIAM J. Optim. 10 (2000) 1079–1096.
[15] J. Moré, S. Wright, Optimization Software Guide, SIAM - Frontiers in Applied Mathematics, Philadelphia, 1993.
[16] S. Nash, Truncated–Newton methods for large-scale function minimization, in: H. Rauch (Ed.), Applications of Nonlinear Programming To Optimization and Control, Pergamon Press, Oxford, 1984, pp. 91–100.
[17] S. Nash, A survey of truncated–Newton methods, J. Comput. Appl. Math. 124 (2000) 45–59.
[18] S. Nash, J. Nocedal, A numerical study of the limited memory BFGS method and the truncated–Newton method for large scale optimization, SIAM J. Optim. 1 (1991) 358–372.
[19] S. Nash, A. Sofer, Assessing a search direction within a truncated–Newton method, Oper. Res. Lett. 9 (1990) 219–221.
[20] S. Nash, A. Sofer, A general–purpose parallel algorithm for unconstrained optimization, SIAM J. Optim. 1 (1991) 530–547.

[21] J. Nocedal, Large scale unconstrained optimization, in: A. Watson, I. Duff (Eds.), The State of the Art in Numerical Analysis, Oxford University Press, Oxford, 1997, pp. 311–338.
[22] T. Schlick, A. Fogelson, TNPACK - A truncated Newton package for large-scale problems: I. Algorithm and usage, ACM Trans. Math. Software 18 (1992) 46–70.
[23] T. Schlick, A. Fogelson, TNPACK - A truncated Newton package for large-scale problems: II. Implementation examples, ACM Trans. Math. Software 18 (1992) 71–111.
[24] D. Xie, T. Schlick, Efficient implementation of the truncated–Newton algorithm for large–scale chemistry applications, SIAM J. Optim. 10 (1999) 132–154.