Armijo-type condition for the determination of a Generalized Cauchy Point in trust region algorithms using exact or inexact projections on convex constraints

by A. Sartenaer*

September 24, 1993

Abstract. This paper considers some aspects of two classes of trust region methods for solving constrained optimization problems. The first class proposed by Toint uses techniques based on the explicitly calculated projected gradient while the second class proposed by Conn, Gould, Sartenaer and Toint allows for inexact projections on the constraints. We propose and analyze, for each class, a step-size rule in the spirit of the Armijo rule for the determination of a Generalized Cauchy Point. We then prove under mild assumptions that, in both cases, the classes preserve their theoretical properties of global convergence and identification of the correct active set in a finite number of iterations. Numerical issues are also discussed for both classes.

* Department of Mathematics, Facultés Universitaires ND de la Paix, 61 rue de Bruxelles, B-5000 Namur, Belgium
Email : as@math.fundp.ac.be

Keywords : Nonlinear optimization, convex constraints, trust region method, projected gradient, Generalized Cauchy Point.

Armijo-type condition for the determination of a Generalized Cauchy Point in trust region algorithms using exact or inexact projections on convex constraints

by A. Sartenaer*

Report 92/13 September 24, 1993

Abstract. This paper considers some aspects of two classes of trust region methods for solving constrained optimization problems. The first class proposed by Toint uses techniques based on the explicitly calculated projected gradient while the second class proposed by Conn, Gould, Sartenaer and Toint allows for inexact projections on the constraints. We propose and analyze, for each class, a step-size rule in the spirit of the Armijo rule for the determination of a Generalized Cauchy Point. We then prove under mild assumptions that, in both cases, the classes preserve their theoretical properties of global convergence and identification of the correct active set in a finite number of iterations. Numerical issues are also discussed for both classes.

* Department of Mathematics, Facultés Universitaires ND de la Paix, 61 rue de Bruxelles, B-5000 Namur, Belgium
Email : as@math.fundp.ac.be

Keywords : Nonlinear optimization, convex constraints, trust region method, projected gradient, Generalized Cauchy Point.

1 Introduction

The *gradient projection* algorithm was originally proposed by Goldstein [12] and Levitin and Polyak [13] for minimizing a continuously differentiable function f from $\mathbf{R}^n$ into $\mathbf{R}$ over a closed convex subset X of $\mathbf{R}^n$. This algorithm is defined by

$$x_{k+1} = P_X(x_k - a_k \nabla f(x_k)), \qquad k = 1, 2, \dots \tag{1.1}$$

where P_X is the projection operator onto X and $a_k \geq 0$ is the step-size. In large scale constrained problems, algorithms based on this method are of great interest, for they allow to drop and add many constraints at each iteration. They are of course especially efficient as the set X is such that the projection can be easily computed.

Among the existing step-size rules for choosing a_k in (1.1), Bertsekas [1] proposed a generalization of the well-known Armijo rule for an efficient and easily implementable variation of the gradient projection method. This type of rule has thereafter been studied by Gafni and Bertsekas [11] and by Dunn (see [10] for instance). Bertsekas [3] also analyzed projected Newton methods using Armijo-type rules for the problem $\min\{f(x)|x \geq 0\}$. More recently, Calamai and Moré [4] generalized the Armijo-type rule proposed by Bertsekas [1] and considered a gradient projection method that selects the step-size a_k using conditions on the model's decrease and the steplength that are in the spirit of Goldstein's rules (see [2] for instance).

Trust region methods for nonlinear optimization problems, another important ingredient of the present paper, are now well-known for their strong numerical reliability and efficiency associated with robust convergence properties (see [14] for a survey on this subject). In an attempt to combine gradient projection and trust region methods, Conn, Gould and Toint [5] first proposed and studied an algorithm for bound constrained problems. Toint [18] and Moré [15] then derived some strong convergence results for algorithms that consider the more general case where the feasible region is a convex set on which the projection is computable at a reasonable cost. However, if the explicitly calculated projected gradient is not available or too costly to obtain, the above methods are useless. For that reason, Conn, Gould, Sartenaer and Toint proposed a class of trust region algorithms that uses inexact projections on convex constraints [7].

As usual in trust region approaches, the algorithms cited in the previous paragraph build, at each iteration k, a model m_k of the objective function at the current feasible point x_k, and a trust region of radius Δ_k around x_k. Inside this trust region, the algorithms first use the first order information to calculate a point that sufficiently reduces the value of the model m_k. This point denoted x_k^C and called *Generalized Cauchy Point* is crucial to ensure the global convergence of the algorithms and is also used to predict which of the convex constraints are satisfied at the problem's solution. The selected Generalized Cauchy Point is then possibly refined by computing a point $x_k + s_k$ based on second order information that provides fast ultimate rate of convergence. The current iterate x_k and the trust region radius Δ_k are then updated using a classical framework based on the ratio of the achieved to the predicted reduction (see [7] and [18]).

To determine a Generalized Cauchy Point (or GCP), theoretical conditions are imposed that are in the spirit of the classical Goldstein conditions for a "projected search" on the model along either the projected gradient path [15], [18], or an approximation of this path [7]. The aim of this paper is to analyze the effect of replacing the Goldstein-like conditions by other rules that are similar in spirit to the efficient Armijo one for the algorithms proposed in [18] and [7]. Note that this issue has already been studied by Moré for its class of algorithms (see [15]).

One motivation behind this analysis is the following: when introducing a particular implementation of the class of algorithms of Toint [18] in their LANCELOT software project [6], Conn, Gould and Toint have chosen to adapt an Armijo-type condition rather than Goldstein-type ones in their code. In order to connect theoretical and practical considerations, we show in this paper under which assumptions the practically used Armijo-type rule implies the Goldstein-type conditions given in [18], so that the global convergence theory of [18] also applies for the implemented version included in LANCELOT. We then further point out the difficulties that arise when trying to generalize the above adaptation to the second class of algorithms of Conn, Gould, Sartenaer and Toint [7], and propose a way to circumvent these obstacles.

Section 2 of the paper studies the first class of algorithms proposed by Toint in [18] while Section 3 considers the second class of algorithms of Conn, Gould, Sartenaer and Toint [7], pointing out the main difficulty for replacing Goldstein-type rules by an Armijo-type rule. Section 4 proposes a new algorithm for the second class and Section 5 analyzes its properties. We discuss, in Section 6, some numerical issues for the given algorithms. Finally conclusions are given in Section 7.

2 An Armijo-type rule for a class of trust region algorithms using the explicit projected gradient path

We first describe more precisely the notations, the assumptions and the class of algorithms considered by Toint in [18]. For convenience, We will denote by $\langle \cdot, \cdot \rangle$ the Euclidean inner product on $\mathbf{R}^n$ and by $\|\cdot\|$ the associated ℓ_2-norm[1]. The convex set X is supposed to be bounded and the gradient $\nabla f(x)$ of the objective function is Lipschitz continuous on an open subset of $\mathbf{R}^n$ containing X. For all k, the model m_k is assumed to be differentiable in an open subset of $\mathbf{R}^n$ containing the trust region and to satisfy the condition

$$m_k(x_k) = f(x_k).$$

Its gradient vector at x_k, $g_k \stackrel{\text{def}}{=} \nabla m_k(x_k)$, approximates the gradient of the objective function at x_k in the following sense: there exist non-negative constants κ_1 and κ_2 such that the inequality

$$\|g_k - \nabla f(x_k)\| \le \min[\kappa_1 \Delta_k, \kappa_2]$$

holds for all k.

If we define the *projected gradient path* by

$$d_k(\theta) \stackrel{\text{def}}{=} P_X(x_k - \theta g_k) - x_k \quad (\theta \ge 0) \tag{2.1}$$

[1] Note that in his paper, Toint considers the more general context of a real Hilbert space.

where P_X is the projection on X, the technique described by Toint uses Goldstein-type rules to calculate a Generalized Cauchy Point at a given iteration k and may be summarized as follows.

Algorithm 1a

Find θ_k^A such that

$$m_k(x_k + d_k(\theta_k^A)) \leq m_k(x_k) + \mu_1 \langle g_k, d_k(\theta_k^A) \rangle, \tag{2.2}$$

$$\|d_k(\theta_k^A)\| \leq \nu_1 \Delta_k, \tag{2.3}$$

and

$$\theta_k^A \geq \nu_2 \theta_k^B \quad \text{or} \quad \theta_k^A \geq \min\{\nu_3 \Delta_k / \|g_k\|, \nu_4\}, \tag{2.4}$$

where θ_k^B (if required) is some strictly positive number that satisfies

$$m_k(x_k + d_k(\theta_k^B)) \geq m_k(x_k) + \mu_2 \langle g_k, d_k(\theta_k^B) \rangle. \tag{2.5}$$

Set the Generalized Cauchy Point

$$x_k^C = x_k + d_k(\theta_k^A).$$

In this description, the constants should satisfy the inequalities

$$0 < \mu_1 < \mu_2 < 1,$$

$$0 < \nu_3 < \nu_1, \quad 0 < \nu_2 \leq 1 \ \text{ and } \ \nu_4 > 0.$$

Since the continuous projected gradient path $d_k(\theta)$ is available, we can replace the Goldstein-type conditions (2.2)–(2.5) by an Armijo-type condition and state the following Generalized Cauchy Point determination algorithm.

Algorithm 1b

Find $\theta_k = \gamma \beta^{n_k}$ where n_k is the first non-negative integer n such that

$$m_k(x_k + d_k(\gamma\beta^n)) \leq m_k(x_k) + \mu_1 \langle g_k, d_k(\gamma\beta^n) \rangle, \tag{2.6}$$

and

$$\|d_k(\gamma\beta^n)\| \leq \nu_1 \Delta_k, \tag{2.7}$$

and where the constants β and γ satisfy $\beta \in (0,1)$ and $\gamma > 0$.

Set the Generalized Cauchy Point

$$x_k^C = x_k + d_k(\theta_k).$$

The first lemma recalls some properties of the quantities $\langle g_k, d_k(\theta) \rangle$ and $\|d_k(\theta)\|$.

Lemma 1 *For all $k \geq 0$,*

1. *the function $\theta \mapsto \|d_k(\theta)\|$ is nondecreasing for $\theta \geq 0$,*

2. if $\|d_k(1)\| > 0$*, then*

$$\langle g_k, d_k(\theta)\rangle \leq -\frac{\|d_k(\theta)\|^2}{\theta} < 0 \tag{2.8}$$

for all $\theta > 0$*, and,*

3. for all $\theta \geq 0$*,*

$$\|d_k(\theta)\| \leq \theta \|g_k\|. \tag{2.9}$$

Proof. The proof of the two first items can be found in Lemma 4 of [18]. We deduce the third item from the non-expansive character of the projection operator,

$$\|P_X(x) - P_X(z)\| \leq \|x - z\| \text{ for all } x, z \in \mathbf{R}^n,$$

for the choice $x = x_k - \theta g_k$ and $z = x_k$, together with the feasibility of x_k and the definition of $d_k(\theta)$ in (2.1). □

Note that, in the above lemma, the non-negative quantity $\|d_k(1)\|$ corresponds to h_k in [18], where it is used as a criticality measure by Toint.

We now prove that the Armijo-type conditions (2.6)–(2.7) imply the Goldstein-type conditions (2.2)–(2.5), with the additional assumption that the gradient of the model m_k at a given iteration k is Lipschitz continuous in an open domain containing the feasible set X.

Theorem 2 *Assume that the gradient of* m_k *is Lipschitz continuous in an open domain containing* X*. Then, if* $\|d_k(1)\| > 0$*, there always exists* $\theta_k > 0$ *satisfying (2.6)–(2.7). Moreover,* θ_k *satisfies conditions (2.2)–(2.5) for the choice*

$$\nu_4 = \gamma, \quad \nu_2 = \beta \text{ and } \nu_3 = \beta\nu_1. \tag{2.10}$$

Proof. The Lipschitz continuity of m_k implies that, for all $\theta \geq 0$,

$$m_k(x_k + d_k(\theta)) \leq m_k(x_k) + \langle g_k, d_k(\theta)\rangle + \frac{1}{2}L_k\|d_k(\theta)\|^2,$$

where L_k is the Lipschitz constant of the gradient of m_k in the norm $\|\cdot\|$. We then deduce from (2.8) that

$$\begin{aligned} m_k(x_k + d_k(\theta)) &\leq m_k(x_k) + \langle g_k, d_k(\theta)\rangle - \tfrac{1}{2}L_k\theta\langle g_k, d_k(\theta)\rangle \\ &= m_k(x_k) + (1 - 1/2L_k\theta)\langle g_k, d_k(\theta)\rangle, \end{aligned}$$

such that

$$m_k(x_k + d_k(\theta)) \leq m_k(x_k) + \mu_1\langle g_k, d_k(\theta)\rangle$$

for all θ satisfying

$$\theta \in [0, \frac{2(1-\mu_1)}{L_k}].$$

On the other hand, the continuity of $d_k(\cdot)$ as a function of θ, the fact that $d_k(\theta) = 0$ for $\theta = 0$ and the nondecreasing nature of $\|d_k(\theta)\|$ imply that

$$\|d_k(\theta)\| \leq \nu_1\Delta_k,$$

for all $\theta \geq 0$ sufficiently small. We thus obtain that $\theta_k = \gamma\beta^{n_k} > 0$ as defined in Algorithm 1b always exists for some $n_k \geq 0$.

Let us now prove that conditions (2.2)–(2.5) are satisfied for $\theta_k = \gamma\beta^{n_k}$. Note that conditions (2.2) and (2.3) obviously hold with $\theta_k^A = \theta_k$. If $n_k = 0$, then $\theta_k = \gamma$ and the result immediately follows from (2.10) and the second part of (2.4). Suppose therefore that $n_k > 0$. In that case, the way in which θ_k is chosen implies that at least one of the two conditions (2.6)–(2.7) must be violated for

$$\theta_k^B \stackrel{\text{def}}{=} \theta_k/\beta = \gamma\beta^{n_k-1}. \tag{2.11}$$

Assume first that condition (2.6) is not satisfied for θ_k^B, that is

$$m_k(x_k + d_k(\theta_k^B)) > m_k(x_k) + \mu_1\langle g_k, d_k(\theta_k^B)\rangle.$$

Then, by (2.10), we have that $\theta_k = \nu_2\theta_k^B$, and the inequality $\mu_1 < \mu_2$ together with the inequality $\langle g_k, d_k(\theta_k^B)\rangle < 0$ from (2.8) imply that θ_k^B satisfies

$$m_k(x_k + d_k(\theta_k^B)) \geq m_k(x_k) + \mu_2\langle g_k, d_k(\theta_k^B)\rangle.$$

The first part of condition (2.4) and condition (2.5) thus hold for $\theta_k^A = \theta_k$ and θ_k^B defined by (2.11).

Suppose now that condition (2.7) is not satisfied for θ_k^B, that is

$$\|d_k(\theta_k^B)\| > \nu_1\Delta_k.$$

In that case, using (2.9) in Lemma 1, we deduce that

$$\theta_k = \beta\theta_k^B \geq \beta\frac{\|d_k(\theta_k^B)\|}{\|g_k\|} \geq \beta\frac{\nu_1\Delta_k}{\|g_k\|} \geq \nu_3\frac{\Delta_k}{\|g_k\|},$$

where the last inequality arises from (2.10). The second part of (2.4) is then satisfied for $\theta_k^A = \theta_k$, which completes the proof.□

This last theorem shows that we can replace Algorithm 1a by Algorithm 1b in the step determination algorithm of [18], without affecting the global convergence results proven by Toint in [18]. As already mentioned, this result was established by Moré for a similar class of algorithms described in [15] (see Theorem 4.2).

3 A class of trust region algorithms using inexact projections on the constraints

In [7], Conn, Gould, Sartenaer and Toint developed a class of trust region algorithms that compute a Generalized Cauchy Point not *on* the projected gradient path but *not far* from it. This was intended to produce algorithms adapted to cases where the projection onto the feasible domain is expensive to calculate. By "not far", we mean a point $x_k^C = x_k + s_k^C$ that satisfies the inequality

$$g_k^T s_k^C \leq -\mu_3\alpha_k(t_k)$$

for some fixed $\mu_3 \in (0,1]$ and $t_k > 0$, where $\alpha_k(t_k) > 0$ represents the magnitude of the maximum decrease of the linearized model achievable on the intersection of the feasible domain with a ball of radius t_k centered at x_k:

$$\alpha_k(t_k) \stackrel{\text{def}}{=} \Big| \min_{\substack{x_k+d\in X \\ \|d\|_{(k)}\le t_k}} \langle g_k, d\rangle \Big|.$$

Here $\|\cdot\|_{(k)}$ denotes the norm associated with iteration k of the algorithm. The norms are indeed allowed to be different at each iteration provided that they are uniformly equivalent (see [7]). Note that the more μ_3 is close to one, the more the candidate for a Generalized Cauchy Point will be near the projected gradient path.

Before describing the class of algorithms, we recall the notations and assumptions used in [7]. The gradient $\nabla f(x)$ of the objective function is assumed to be Lipschitz continuous in an open domain containing $\mathcal{L}$, where $\mathcal{L} \stackrel{\text{def}}{=} X \cap \{x \in \mathbf{R}^n \mid f(x) \le f(x_0)\}$ is supposed to be compact and x_0 is a feasible starting point. The model m_k has to be differentiable, to have Lipschitz continuous first derivatives in an open subset containing the trust region and to satisfy the condition

$$m_k(x_k) = f(x_k).$$

Finally, its gradient vector at x_k, $g_k \stackrel{\text{def}}{=} \nabla m_k(x_k)$, approximates $\nabla f(x_k)$ in the following sense: there exists a non-negative constant κ_1 such that the inequality

$$\|g_k - \nabla f(x_k)\|_{[k]} \le \kappa_1 \Delta_k$$

holds for all k, where the norm $\|\cdot\|_{[k]}$ is any norm that satisfies

$$|\langle x, y\rangle| \le \|x\|_{(k)}\, \|y\|_{[k]}$$

for all $x, y \in \mathbf{R}^n$.

The theoretical conditions on a Generalized Cauchy Point in [7] are the following.

Algorithm 2

Find a vector s_k^C such that, for some strictly positive $t_k \ge \|s_k^C\|_{(k)}$,

$$x_k + s_k^C \in X, \tag{3.1}$$

$$\|s_k^C\|_{(k)} \le \nu_1 \Delta_k, \tag{3.2}$$

$$\langle g_k, s_k^C\rangle \le -\mu_3 \alpha_k(t_k), \tag{3.3}$$

$$m_k(x_k + s_k^C) \le m_k(x_k) + \mu_1 \langle g_k, s_k^C\rangle, \tag{3.4}$$

and

$$t_k \ge \min\{\nu_3 \Delta_k, \nu_4\} \;\text{ or }\; m_k(x_k + s_k^C) \ge m_k(x_k) + \mu_2 \langle g_k, s_k^C\rangle. \tag{3.5}$$

Set the Generalized Cauchy Point

$$x_k^C = x_k + s_k^C.$$

Here the constants satisfy

$$0 < \mu_1 < \mu_2 < 1, \quad \mu_3 \in (0,1],$$

$$0 < \nu_3 < \nu_1 \text{ and } \nu_4 \in (0,1].$$

The *GCP Algorithm* given in [7] is a *practical model* algorithm for computing a Generalized Cauchy Point that is coherent with the theoretical framework of Algorithm 2. This algorithm is iterative and proceeds by bisection. At each iteration i, given a bisection parameter value $t_i > 0$, it computes first a candidate step s_i that satisfies condition (3.3) (with $s_k^C = s_i$ and $t_k = t_i$). It then checks conditions (3.4) and (3.5) until either an acceptable GCP is found or two candidates $x_k + s_k^l$ and $x_k + s_k^u$ are known that violate, one the second part of condition (3.5), the other condition (3.4). In this last case, the GCP Algorithm calls the *RS Algorithm* that carries out a simple bisection linesearch on the model along a particular path between these two points, yielding a suitable GCP in a finite number of iterations. This particular path, called the *restricted path*, is obtained by applying the so-called *restriction operator*,

$$R_{x_k}[y] \stackrel{\text{def}}{=} \arg \min_{z \in [x_k, y] \cap X} \|z - y\|_2$$

where $[x_k, y]$ is the segment between x_k and y, on the piecewise linear path consisting of the segment $[x_k + s_k^l, x_k + s_k^p]$ followed by $[x_k + s_k^p, x_k + s_k^u]$, where

$$s_k^p = \max \left[1, \frac{\|s_k^u\|_{(k)}}{\|s_k^l\|_{(k)}} \right] s_k^l.$$

This restricted path is an approximation of the unknown projected gradient path between the points $x_k + s_k^l$ and $x_k + s_k^u$ in the sense that each point on this path satisfies condition (3.3) for some $t_k > 0$. It also closely follows the boundary of the feasible domain, as does the projected gradient path. We refer the reader to [7] for more details.

The RS Algorithm can thus be applied under the conditions that the second part of (3.5) is violated at $R_{x_k}[x_k + s_k^l] = x_k + s_k^l$ and that (3.4) is violated at $R_{x_k}[x_k + s_k^u] = x_k + s_k^u$. It therefore depends on the three points $x_k + s_k^l$, $x_k + s_k^p$ and $x_k + s_k^u$ defining the piecewise linear path, on the centre x_k, and on the current model m_k (and hence on its gradient g_k). Lemma 20 in [7] shows that this algorithm terminates with a suitable point at which the second part of condition (3.5) and condition (3.4) hold in a finite number of iterations.

As in Section 2 for the case of exact projections, we would like to reshape Algorithm 2 using an Armijo-type condition, making valid the theoretical properties proved in [7]. Note however that this is not a straightforward task, mainly because the Armijo-type rule usually requires the search on the model to be performed along a given direction, or at least a given path (as it is the case for [18]). The difficulty here arises from the non-existence of such a direction (or path) since Algorithm 2 only generates points that are "not too far from the unavailable projected gradient path" through condition (3.3). We will thus have to use some ingenuity in order to include a rule which is in the spirit of the Armijo one and restate the GCP Algorithm of [7], rather than Algorithm 2 itself, in an attempt to generalize Theorem 2 of Section 2.

4 An Armijo-type rule for the class of trust region algorithms of Section 3

The reason why the second part of Theorem 2 in Section 2 holds is largely due to the fact that condition (2.5) is allowed to be satisfied by another step than $d_k(\theta_k^A)$, namely $d_k(\theta_k^B)$, provided that this step satisfies $\theta_k^A \geq \nu_2 \theta_k^B$. Therefore, as soon as a step $d_k(\gamma\beta^{n_k})$ satisfying the Armijo-type conditions (2.6) and (2.7) has been found, it is then easy to deduce the existence of another step, $d_k(\gamma\beta^{n_k-1})$, satisfying the first part of (2.4) and (2.5) (see the proof of Theorem 2).

Unfortunately, it is not possible to impose conditions analogous to the first part of (2.4) and (2.5) in the framework of Algorithm 2, as the step s_k^C to be found does not belong to a well-known path. It rather has to satisfy condition (3.3) for which the set of suitable solutions for different values of t_k does not define a single path. For that reason, we need here to adapt the Armijo-type rule in order to derive the counterpart of Theorem 2 for a *particular model* of Algorithm 2, related to the GCP Algorithm.

This particular model, denoted the *GCP$_{\mathcal{A}}$ Algorithm*, depends on the current iterate $x_k \in X$, on the current model m_k and its gradient g_k, on the current norm $\|\cdot\|_{(k)}$ and also on the current trust region radius, $\Delta_k > 0$. It also depends on some scalars $\gamma > 0$ and $\beta \in (0,1)$. Its inner iterations are identified by the index i.

GCP$_{\mathcal{A}}$ Algorithm

Step 0 : initialization. Set $i = 0$. Choose s_0^u an arbitrary vector such that $\|s_0^u\|_{(k)} > \nu_1 \Delta_k$ and an initial parameter $t_0 = \gamma$, for some $\gamma > 0$. Also choose $\beta \in (0,1)$.

Step 1 : compute a candidate step. Compute a vector s_i such that

$$\|s_i\|_{(k)} \leq t_i,$$

$$x_k + s_i \in X,$$

and

$$\langle g_k, s_i \rangle \leq -\mu_3 \alpha_k(t_i). \tag{4.1}$$

Step 2 : check the Armijo-type stopping rule on the model. If

$$m_k(x_k + s_i) > m_k(x_k) + \mu_1 \langle g_k, s_i \rangle, \tag{4.2}$$

then set

$$s_{i+1}^u = s_i, \tag{4.3}$$

and

$$t_{i+1} = \beta t_i,$$

increment i by one and go to Step 1. Else, go to Step 3.

Step 3 : compute a GCP. If

$$m_k(x_k + s_i) < m_k(x_k) + \mu_2\langle g_k, s_i\rangle, \tag{4.4}$$

and

$$t_i < \min\{\nu_3\Delta_k, \nu_4\}, \tag{4.5}$$

then set

$$s_k^l = s_i \text{ and } s_k^u = s_i^u, \tag{4.6}$$

define

$$s_k^p = \max\left[1, \frac{\|s_k^u\|_{(k)}}{\|s_k^l\|_{(k)}}\right] s_k^l, \tag{4.7}$$

apply the RS Algorithm with $x_k + s_k^l$, $x_k + s_k^p$ and $x_k + s_k^u$ and STOP. Else (that is if (4.2) and either (4.4) or (4.5) fail), then set

$$x_k^C = x_k + s_i, \tag{4.8}$$

and STOP.

Note that the update (4.3) in Step 2 is intended only to produce a well-defined step s_k^u whenever it is needed. The next lemma shows that the above GCP$_\mathcal{A}$ Algorithm is correctly stated, provided the following restriction on the possible choices of γ holds

$$\gamma \in [\min\{\nu_3\Delta_k, \nu_4\}, \nu_1\Delta_k]. \tag{4.9}$$

Lemma 3 *Assume that (4.9) holds. Then the GCP$_\mathcal{A}$ Algorithm has well-defined iterates.*

Proof. We have to verify that all the required conditions for applying the RS Algorithm are fulfilled when a call to this algorithm is made. For that, we distinguish two cases. Assume first that (4.2) is violated for $i = 0$. In that case, (4.9) guarantees that $t_0 = \gamma$ also violates condition (4.5), such that the algorithm immediately terminates with (4.8). Therefore, the RS Algorithm may be called only at some iteration i where $i > 0$. But then the GCP$_\mathcal{A}$ Algorithm has passed at least once through (4.3) and ensures, together with (4.6) and (4.7), that the RS Algorithm is applied with well-defined vectors $x_k + s_k^l$, $x_k + s_k^p$ and $x_k + s_k^u$. Moreover, the mechanism of the GCP$_\mathcal{A}$ Algorithm ensures that the piecewise path to be restricted is non-empty, that (3.4) is always violated at $R_{x_k}[x_k + s_k^u] = x_k + s_k^u$ and similarly that the second part of (3.5) is always violated at $R_{x_k}[x_k + s_k^l] = x_k + s_k^l$. We thus deduce that the RS Algorithm is applied in the appropriate context by finally noting that it can only produce a feasible point because of the definition of the restriction operator. □

Note that the restriction (4.9) on γ is actually due to the fact that the rule used in the GCP$_\mathcal{A}$ Algorithm is of backtracking type, so that we need to guarantee that the initial parameter $t_0 = \gamma$ (and thus the first step selected) is sufficiently large, while comparable to the trust region radius.

5 Properties of the GCP$_\mathcal{A}$ Algorithm

Like for the GCP Algorithm in [7], we wish to show that the GCP$_\mathcal{A}$ Algorithm is finite and coherent with the theoretical framework presented in Algorithm 2 (that is, converges to a point satisfying (3.1)–(3.5)). We first prove the desirable finiteness of the GCP$_\mathcal{A}$ Algorithm at non-critical points. Note that the quantity $\alpha_k \stackrel{\rm def}{=} \alpha_k(1)$ mentioned in Theorem 4, where

$$\alpha_k(t) \stackrel{\rm def}{=} |\min_{\substack{x_k+d\in X\\ \|d\|_{(k)}\le t}} \langle g_k, d\rangle|,$$

corresponds to the criticality measure of [7]. Moreover, we shall make use of the following property

$$\alpha_k(t) \ge t\alpha_k \;\text{ for all }\; 0\le t\le 1, \tag{5.1}$$

which is an easy consequence of Lemma 2.2 in [7].

Theorem 4 *Assume that (4.9) holds and that $\alpha_k > 0$. Then the GCP$_\mathcal{A}$ Algorithm terminates with a suitable x_k^C in a finite number of iterations.*

Proof. Assume, for the purpose of obtaining a contradiction, that an infinite number of iterations are performed. Note first that Lemma 20 in [7] ensures finite termination in the case where the RS Algorithm is used. Therefore, the mechanism of the algorithm implies that, for the algorithm to be infinite, one must have that (4.2) is satisfied for all $i\ge 0$, and that

$$t_i = \beta^i\gamma$$

for all $i\ge 0$, where $\beta\in(0,1)$. Hence we obtain, using the inequalities $\alpha_k>0, \mu_1<1$ and $\mu_3>0$, that

$$\|s_i\|_{(k)} \le t_i \le \min\left[1, \frac{2(1-\mu_1)\mu_3\alpha_k}{L_k}\right] \tag{5.2}$$

for all $i\ge i_1$, say, where L_k is the Lipschitz constant of the gradient of m_k with respect to the norm $\|\cdot\|_{(k)}$.

On the other hand, for all $i\ge 0$, we have that

$$m_k(x_k+s_i) - m_k(x_k) - \mu_1\langle g_k, s_i\rangle \le (1-\mu_1)\langle g_k, s_i\rangle + \frac{1}{2}L_k\|s_i\|^2_{(k)}, \tag{5.3}$$

where we have used the Taylor's expansion of m_k around x_k and the definition of L_k. But (5.1) and (5.2) imply that

$$\alpha_k(t_i) \ge t_i\alpha_k$$

for all $i\ge i_1$, and hence, using (5.2) again, we deduce that

$$\alpha_k(t_i) \ge \alpha_k\|s_i\|_{(k)}$$

for $i\ge i_1$. Condition (4.1) then gives, for such i, that

$$\langle g_k, s_i\rangle \le -\mu_3\alpha_k(t_i) \le -\mu_3\alpha_k\|s_i\|_{(k)}.$$

Introducing this inequality in (5.3), we obtain that

$$m_k(x_k+s_i) - m_k(x_k) - \mu_1\langle g_k, s_i\rangle \le -(1-\mu_1)\mu_3\alpha_k\|s_i\|_{(k)} + \frac{1}{2}L_k\|s_i\|^2_{(k)},$$

for $i \ge i_1$. Using (5.2), we now deduce that

$$m_k(x_k+s_i) - m_k(x_k) - \mu_1\langle g_k, s_i\rangle \le 0$$

for all $i \ge i_1$. As a consequence, (4.2) is always violated for sufficiently large i and we obtain the desired contradiction. □

Let us now prove the consistency of the GCP$_{\mathcal{A}}$ Algorithm with the theoretical framework of Algorithm 2.

Theorem 5 *Assume that (4.9) holds. Then the GCP$_{\mathcal{A}}$ Algorithm can be used as an implementation of Algorithm 2.*

Proof. We have to verify the compatibility of the GCP$_{\mathcal{A}}$ Algorithm with the conditions of Algorithm 2, that is we have to check that the step $s_k^C = x_k^C - x_k$ produced by the GCP$_{\mathcal{A}}$ Algorithm does indeed satisfy conditions (3.1)–(3.5). All these conditions except (3.3) are clearly enforced by the mechanism of the GCP$_{\mathcal{A}}$ and RS Algorithms. We can therefore restrict our attention to the verification of (3.3) for the two different possible exits of the GCP$_{\mathcal{A}}$ Algorithm and their associated $s_k^C = x_k^C - x_k$.

The first case is when the GCP$_{\mathcal{A}}$ Algorithm terminates using (4.8). Then (4.1) ensures that (3.3) holds for $s_k^C = s_i$ and $t_k = t_i$.

The second and last case is when the algorithm terminates using the RS Algorithm. The condition (4.1) again ensures that, in this case, (3.3) holds for $s_k^C = s_k^l$ and some $t_k = t^l \ge \|s_k^l\|_{(k)}$, and for $s_k^C = s_k^u$ and some $t_k = t^u \ge \|s_k^u\|_{(k)}$. We are then exactly in the same position than for the GCP Algorithm. Consequently, the result follows from Theorem 24 of [7], which establishes that relation (3.3) is satisfied by every point on the restricted path. □

According to Theorems 4 and 5, the GCP$_{\mathcal{A}}$ Algorithm may be considered as a particular model algorithm of the theoretical Algorithm 2, so that the properties proved in [7] remain valid, especially the global convergence and the identification of the correct active set in a finite number of iterations.

6 Numerical experiments

In [8] and [9], Conn, Gould and Toint report on intensive numerical experiments with the LANCELOT code (Release A). This code proposes a list of 21 different algorithmic choices (see [9]). Among those, the Armijo condition of Algorithm 1b is implemented under the name appGCP (for *approximate Generalized Cauchy Point*). This variant is to be compared with the **default** algorithmic choice that uses an exact Generalized Cauchy Point calculation, that is the calculation of $x_k + d_k(\theta_e)$ where θ_e is the first minimizer of $m_k(x_k + d_k(\theta_e))$ subject to the trust region constraints (2.3). Note that this exact calculation may be very costly compared with an Armijo-type approach, especially in the presence of general nonlinear convex constraints.

The analysis in [9] based on computational tests shows that the appGCP variant decreases the reliability compared with the default choice of an exact computation. This is to be expected, given the less exacting conditions on a Generalized Cauchy Point imposed by the Armijo-type approach. However, as pointed out in [9], this decrease in reliability is not dramatical. We can even say that the cputimes are more than occasionally better for the appGCP variant on large problems—see for instance problems BRITGAS, CLPLATEC (5041), DALLASL, DIXMAANF (3000), DQRTIC (5000), GRIDNETD (3444), HYDROELM, JNLBRNGB (15625), NONDIA (10000), PENALTY1 (1000) and SVANBERG (1000) in [8].

In [17], we have adapted the class of trust region algorithms developed in [7] to the nonlinear network problem. The resulting algorithm, that uses the GCP Algorithm for the computation of a Generalized Cauchy Point, is fully described and discussed on the light of numerical experimentations. In order to make comparisons between Goldstein-type and Armijo-type conditions, we then have implemented the $\mathrm{GCP}_{\mathcal{A}}$ Algorithm of Section 4 to replace the GCP Algorithm in the solution of the nonlinear network problem. Note that for both methods, we have used the same values for the algorithms' constants (see [17]) and we have performed all the computations in double precision on a DEC VAX 3500, under VMS, using the standard Fortran Compiler ($\epsilon_M \simeq 1.39 \times 10^{-17}$). The results, reported in [16], show that the performance in term of number of iterations are identical for both the GCP and $\mathrm{GCP}_{\mathcal{A}}$ Algorithms on all the problem tests performed (about 150), except for one, for which the difference observed was very slight. This may be explained by the practical choice of a sufficiently big initial parameter t_0 in the GCP and the $\mathrm{GCP}_{\mathcal{A}}$ Algorithms, namely

$$t_0 = \tau_0 \ \text{ and } \ t_0 = \max\left[\min(10^{-5}\Delta_k, 0.01), \tau_0\right]$$

where

$$\tau_0 = \min\left[\left|\frac{\langle g_k, g_k\rangle}{\langle g_k, \nabla^2 f(x_k) g_k\rangle}\right| \|g_k\|_\infty, \Delta_k\right],$$

respectively, that produces the same iterations for both algorithms. On the execution time point of vue, we did not observe significant differences, considering the relative error margin due to the busy character of the time-shared machine. We therefore conclude that, for the nonlinear network application, an Armijo-type condition is at least competitive with a Goldstein-type condition, while both much easier to understand and to implement.

7 Conclusions

In this paper, we propose some modifications of two classes of trust region algorithms using either exact [18] or inexact [7] projections on convex constraints. These modifications intend to replace Goldstein-type conditions in the search of a Generalized Cauchy Point by an Armijo-type rule so as to preserve the theoretical results of [18] and [7].

This analysis completes the study on the use of Armijo-type rules in gradient projection methods carried out by Bertsekas [1] and Dunn [10], for instance, by making the link with trust region strategies.

It further ensures a theoretical support to the numerical results produced by LANCELOT [6] and by the code based on the GCP$_{\mathcal{A}}$ Algorithm described in [16].

Moreover, the numerical experiments performed on nonlinear network problems in [16] and [17] show that the GCP$_{\mathcal{A}}$ Algorithm is clearly comparable in performance with the GCP Algorithm, while being easier to implement.

We finally stress on the fact that the analysis in the present paper has been done preserving the context of [18] and [7]. However, this analysis does actually not depend on the trust region models m_k and on the iteration numbers k of the outer iterations. So, the model m_k could be replaced by a general function f having the required smoothness properties throughout the given theory. We may then conclude in saying that the Armijo methods of Algorithms 1b and GCP$_{\mathcal{A}}$ satisfy the Goldstein conditions (2.2)–(2.5) and (3.1)–(3.5) when the constraints on the variables are achieved by exact or inexact projections onto the feasible region, respectively.

8 Acknowledgement

The author wish to thank Philippe Toint for his useful comments and suggestions.

References

[1] D. P. Bertsekas. On the Goldstein-Levitin-Polyak gradient projection method. *IEEE Transactions on Automatic Control*, 21:174–184, 1976.

[2] D. P. Bertsekas. *Constrained Optimization and Lagrange Multipliers Methods.* Academic Press, London, 1982.

[3] D. P. Bertsekas. Projected Newton methods for optimization problems with simple constraints. *SIAM Journal on Control and Optimization*, 20(2):221–246, 1982.

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

[5] 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.

[6] 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.

[7] 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.

[8] A. R. Conn, Nick Gould, and Ph. L. Toint[1]. Intensive numerical tests with lancelot (Release A): the complete results. Technical Report 92/15, Department of Mathematics, FUNDP, Namur, Belgium, 1992. Also issued as Research Report RC 18750, IBM T.J. Watson Center, Yorktown Heights, USA, and as Research Report 92-069, RAL, Chilton, UK.

[9] A. R. Conn, Nick Gould, and Ph. L. Toint[1]. Numerical experiments with the lancelot package (Release A) for large-scale nonlinear optimization. Technical Report 92/16, Department of Mathematics, FUNDP, Namur, Belgium, 1992. Also issued as Research Report RC 18434, IBM T.J. Watson Center, Yorktown Heights, USA, and as Research Report 92-075, RAL, Chilton, UK.

[10] J. C. Dunn. Global and asymptotic convergence rate estimates for a class of projected gradient process. *SIAM Journal on Control and Optimization*, 19:368–400, 1981.

[11] E. M. Gafni and D. P. Bertsekas. Two-metric projection methods for constrained optimization'. *SIAM Journal on Control and Optimization*, 22:936–964, 1984.

[12] A. A. Goldstein. Convex programming in hilbert space. *Bulletin of the American Mathematical Society*, 70:709–710, 1964.

[13] E. S. Levitin and B. T. Polyak. Constrained minimization problems. *USSR Comput. Math. and Math. Phys.*, 6:1–50, 1966.

[14] 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.

[15] 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.

[16] A. Sartenaer. *On some strategies for handling constraints in nonlinear optimization.* PhD thesis, Department of Mathematics, FUNDP, Namur, Belgium, 1991.

[17] A. Sartenaer[1]. A class of trust region methods for nonlinear network optimization problems, including some numerical experiments (revised). Technical Report 93/21, Department of Mathematics, FUNDP, Namur, Belgium, 1993.

[18] 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.

[1] These reports are available by anonymous ftp from the directory "reports" on thales.math.fundp.ac.be (internet 138.48.4.14)