A Global Convergence Theory For General Trust-Region-Based Algorithms For Equality Constrained Optimization

J. E. Dennis, Jr., Mahmoud El-Alem,
& Maria C. Maciel

CRPC-TR92247
September 1992

Center for Research on Parallel Computing
Rice University
P.O. Box 1892
Houston, TX 77251-1892

Revised September 10, 1993.

A GLOBAL CONVERGENCE THEORY FOR GENERAL TRUST-REGION-BASED ALGORITHMS FOR EQUALITY CONSTRAINED OPTIMIZATION [†]

J. E. DENNIS, JR. [‡], MAHMOUD EL-ALEM[§], AND MARIA C. MACIEL[¶]

Abstract. This work presents a global convergence theory for a broad class of trust-region algorithms for the smooth nonlinear programming problem with equality constraints. The main result generalizes Powell's 1975 result for unconstrained trust-region algorithms.

The trial step is characterized by very mild conditions on its normal and tangential components. The normal component need not be computed accurately. The theory requires a relaxed normal component to satisfy a fraction of Cauchy decrease condition on the quadratic model of the linearized constraints. The tangential component then must satisfy a fraction of Cauchy decrease condition on a quadratic model of the Lagrangian function in the translated tangent space of the constraints determined by the relaxed normal component. The Lagrange multipliers estimates and the Hessian estimates are assumed only to be bounded.

The other main characteristic of this class of algorithms is that the step is evaluated by using the augmented Lagrangian as a merit function and the penalty parameter is updated using the El-Alem scheme. The properties of the step together with the way that the penalty parameter is chosen are sufficient to establish global convergence.

As an example, an algorithm is presented which can be viewed as a generalization of the Steihaug-Toint dogleg algorithm for the unconstrained case. It is based on a quadratic programming algorithm that uses a step in a relaxed normal direction to the tangent space of the constraints and then does feasible conjugate reduced-gradient steps to solve the quadratic program. This algorithm should cope quite well with large problems for which effective preconditioners are known.

Key Words: Constrained Optimization, Global Convergence, Trust Regions, Equality Constrained, Nonlinear Programming, Conjugate Gradient, Inexact Newton Method.

AMS subject classifications. 65K05, 49D37.

1. Introduction. This work is concerned with the development of a global convergence theory for a broad class of algorithms for the equality constrained minimization problem:

$$(\text{EQC}) \equiv \begin{cases} \textit{minimize} & f(x) \\ \textit{subject to} & C(x) = 0. \end{cases}$$

The functions $f : \Re^n \to \Re$ and $C : \Re^n \to \Re^m$ are at least twice continuously differentiable where $C(x) = (c_1(x), \ldots, c_m(x))^T$ and $m < n$.

Our purpose is to generalize to constrained problems a powerful theorem given in 1975 by Powell for the unconstrained problems.

The global convergence theory that we establish in this work holds for a class of nonlinear programming algorithms for (EQC) that is characterized by the following features:

1. The algorithms of the family use the *trust-region approach* as a globalization strategy.

[†] Research was supported by DOE DE-FG005-86ER25017, CRPC CCR-9120008, AFOSR-F49620-9310212, ARO-DAAL03-90-G0093, and the REDI Foundation.

[‡] Department of Computational and Applied Mathematics, Rice University, P. O. Box 1892, Houston TX 77251.

[§] Department of Mathematics, Faculty of Science, Alexandria University, Alexandria, Egypt.

[¶] Departamento de Matematica, Universidad Nacional del Sur, Avenida Alem 1253, 8000 Bahia Blanca, Argentina.

2. All these algorithms generate steps that satisfy very mild conditions on the trial steps' normal and tangential components. The normal component satisfies a *fraction of Cauchy decrease* condition on the quadratic model of the linearized constraints. It is important to note that the normal component s_c^n is not required to be strictly normal. Instead, it is allowed to satisfy the relaxed condition that $\|s_c^n\| \leq K\|C(x_c)\|$ for some independent constant K. The tangential component satisfies a *fraction of Cauchy decrease* on the quadratic model of the reduced Lagrangian function associated with (EQC).
3. The estimates of the Lagrange multiplier vector and the Hessian matrix are assumed only to be bounded uniformly across all iterations.
4. The other main characteristic of this class of algorithms is that the step is evaluated for acceptance by using the augmented Lagrangian function with penalty parameter updated by the scheme proposed by El-Alem [8].

Conditions 1, 2, and 3 are satisfied by the algorithms of Byrd, Schnabel, and Shultz [2], Celis, Dennis, and Tapia [4], Omojokun [18], and Powell and Yuan [21].

We use the following notation: the sequence of points generated by an algorithm is denoted by $\{x_k\}$. This work also uses subscripts -, c and + to denote the previous, the current and the next iterates respectively. However, when we need to work with a whole sequence we will use the index k. The matrix H_c denotes the Hessian of the Lagrangian at the current iterate or an approximation to it. Subscripted functions mean the function is evaluated at a particular point; for example, $f_c = f(x_c)$, $\ell_c = \ell(x_c, \lambda_c)$, and so on. Finally, unless otherwise specified, all the norms will be ℓ_2-norms.

The rest of the paper is organized as follows: In Section 2, we review the concept of fraction of Cauchy decrease. In Section 3, we review the SQP algorithm. In Section 4, we survey existing trust-region algorithms for solving problem (EQC). In Section 5, we present a general trust-region algorithm with the conditions that the trial step must satisfy. In Section 6 we state the algorithm. Sections 7 and 8 are devoted to presenting the global convergence theory that we have developed. In Section 7.1, we state the assumptions under which global convergence is established. In Section 7.2, we discuss some properties of the trial steps. In Section 7.3, we study the behavior of the penalty parameter. Section 8 is devoted to presenting our main global convergence result. In Section 9, we present, as an example, an algorithm that solves problem (EQC), and we prove that it fits the assumptions of the paper. This algorithm can be viewed as a generalization to constrained case of the Steihaug-Toint dogleg algorithm for the unconstrained case. This algorithm has worked quite well for some large problems. Finally, we make some concluding remarks in Section 10.

2. Fraction of Cauchy decrease condition.

Consider the following unconstrained minimization problem

$$\text{(UCMIN)} \equiv \begin{cases} \textit{minimize} & f(x) \\ \textit{subject to} & x \in \Re^n, \end{cases}$$

where $f : \Re^n \rightarrow \Re$ is a continuously differentiable function. A trust-region algorithm for solving the above problem is an iterative procedure that computes ***a trial step*** as an approximate solution to the following ***trust-region subproblem:***

$$\text{(TRS)} \equiv \begin{cases} \textit{minimize} & m_c(s) = f_c + \nabla f_c^T s + \frac{1}{2} s^T G_c s \\ \textit{subject to} & \|s\|_2 \leq \delta_c, \end{cases}$$

where G_c is the Hessian matrix $\nabla^2 f_c$ or an approximation to it and $\delta_c > 0$ is a given trust-region radius. For complete survey see Moré [15] and the book of Dennis and

Schnabel [6].

To assure global convergence, the step is required only to satisfy a *fraction of Cauchy decrease* condition. This means that s_c must predict via the quadratic model function m_c at least as much as a fraction of the decrease given by the Cauchy step on m_c, that is, there exists a constant $\sigma > 0$ fixed across all iterations, such that

$$m_c(0) - m_c(s_c) \;\geq\; \sigma[m_c(0) - m_c(s_c^{\mathrm{cp}})], \tag{2.1}$$

where $s_c^{\mathrm{cp}} = -t_c^{\mathrm{cp}} \nabla f_c$ and its step length

$$t_c^{\mathrm{cp}} = \begin{cases} \frac{\|\nabla f_c\|^2}{\nabla f_c^T G_c \nabla f_c} & \text{if } \frac{\|\nabla f_c\|^3}{\nabla f_c^T G_c \nabla f_c} \leq \delta_c \text{ and } \nabla f_c^T G_c \nabla f_c > 0 \\ \frac{\delta_c}{\|\nabla f_c\|} & \text{otherwise.} \end{cases}$$

Thus, s_c^{cp} is the steepest descent step for m_c inside the trust region.

The form of (2.1) we use to prove convergence is given in the following technical lemma. More details about the role of this lemma in the convergence theory of trust-region algorithms can be found in Carter [3], Moré [15], Powell [20], and Shultz, Schnabel and Byrd [23].

LEMMA 2.1. *If the trial step s_c satisfies a fraction of Cauchy decrease condition, then*

$$m_c(0) - m_c(s_c) \;\geq\; \frac{\sigma}{2} \|\nabla f_c\| \min\left\{ \frac{\|\nabla f_c\|}{\|G_c\|},\, \delta_c \right\}. \tag{2.2}$$

Proof. See Powell [20]. □

We end this section by stating Powell's powerful theorem for unconstrained trust-region algorithms. The proof can be found in Powell [20]. More details about the convergence theory for trust-region algorithms for unconstrained optimization can be found in Fletcher [11], Moré [15], Moré and Sorensen [16], and Sorensen [24].

THEOREM 2.2. *Let $f : \Re^n \to \Re$ be continuously differentiable and bounded below on the level set $\{x \in \Re^n : f(x) \leq f(x_0)\}$. Assume that the sequence $\{G_k\}$ is uniformly bounded. If $\{x_k\}$ is the sequence generated by any trust-region algorithm that satisfies (2.1) or (2.2), then:*

$$\liminf_{k \to \infty} \|\nabla f_k\| = 0.$$

3. The SQP algorithm. The Lagrangian function $\ell : \Re^n \times \Re^m \to \Re$ associated with problem (EQC) is the function

$$\ell(x, \lambda) = f(x) + \lambda^T C(x),$$

where $\lambda = (\lambda_1, ..., \lambda_m)^T$ is a Lagrange multiplier vector estimate.

A common algorithm for solving problem (EQC) is the successive quadratic programming algorithm. It is an iterative procedure. At each iteration, a step s^{QP} and associated Lagrange multiplier $\Delta\lambda^{QP}$ are obtained by solving the following quadratic program

$$(\mathrm{QP}) \equiv \begin{cases} \text{minimize} & q_c(s) = \frac{1}{2} s^T H_c s + \nabla_x \ell_c^T s + \ell_c \\ \text{subject to} & \nabla C_c^T s + C_c = 0, \end{cases}$$

where the matrix H_c is the Hessian of the Lagrangian at (x_c, λ_c) or an approximation to it.

Unfortunately, the SQP algorithm can not be guaranteed to work without modification. There is a fundamental difficulty in the definition of the SQP step because the second-order sufficiency condition need not hold at each iteration. By this we mean that, the matrix H_c need not be positive definite on the null space of ∇C_c^T; hence the QP subproblem may not have a solution or a unique solution. This difficulty will not arise near a solution of problem (EQC) if the standard assumptions for Newton's method hold at the solution. For this reason, the SQP algorithm usually performs very well locally. See Tapia [26] for more details.

An effective modification that deals with the lack of positive definiteness on the null space is to use a trust-region globalization strategy. This takes us to the following section.

4. Existing trust-region algorithms for (EQC). A straightforward way to extend the trust-region idea to problem (EQC) is to add a trust-region constraint to the (QP) subproblem to restrict the size of the step. So, at each iteration, we solve the following trust-region subproblem:

$$\left\{\begin{array}{ll} \textit{minimize} & q_c(s) = \frac{1}{2}s^T H_c s + \nabla_x \ell_c^T s + \ell_c \\ \textit{subject to} & \nabla C_c^T s + C_c = 0 \\ & \|s\| \le \delta_c. \end{array}\right.$$

However, in this straightforward approach, observe that the trust-region constraint and the linearized constraints may be inconsistent, and thus the model subproblem will not have a solution. To overcome this difficulty, two main approaches have been introduced for dealing with the case when $\{s : \nabla C_c^T s + C_c = 0\} \cap \{s : \|s\| \le \delta_c\} = \emptyset$. They are the tangent-space approach, and the full-space approach. We describe them briefly in the next section. More details can be found in Maciel [14]. See also Byrd, Schnabel and Shultz [2], Celis, Dennis and Tapia [4], Omojokun [18], Powell and Yuan [21], and Vardi [29] and [30].

4.1. The tangent-space approach. In this approach the trial step is determined as $s_c = s_c^n + s_c^t$ where s_c^n is the normal component, that is s_c^n is inside the trust region and in the normal direction to the null-space of the constraint Jacobian, $\mathcal{N}(\nabla C_c^T)$, and s_c^t is the component of the step in the tangent-space of the constraints given by $s_c^t = W_c \bar{s}_c^t$, with $\bar{s}_c^t \in \Re^{n-m}$ and W_c is an $n \times (n-m)$ matrix whose columns form a basis for $\mathcal{N}(\nabla C_c^T)$.

This gives two questions to be answered. We must say how to determine s_c^n, and given s_c^n, we must say how to determine s_c^t. We proceed in reverse order. Given s_c^n, we determine s_c^t by considering the transformed subproblem

$$\left\{\begin{array}{ll} \textit{minimize} & q_c(s^t + s_c^n) \\ \textit{subject to} & \nabla C_c^T s^t = 0 \\ & \|s^t\| \le \bar{\delta}_c, \end{array}\right.$$

where $\bar{\delta}_c = \sqrt{\delta_c^2 - \|s_c^n\|^2}$. We choose s_c^t by using one of the standard unconstrained trust-region trial-step selection methods on this reduced problem.

These algorithms have the trust region capability of dealing quite well with zero or negative curvature in the tangent space of constraints. Thus, nonexistence of an SQP step at the current iterate is readily handled.

To choose s_c^n, Byrd, Schnabel and Shultz [2] and Vardi [29],[30] suggest relaxing the linearized constraints by replacing C_c by αC_c where $\alpha \in (0, 1]$, is chosen to ensure that the above trust-region subproblem is feasible. Thus, $s_c^n = -\alpha(\nabla C_c^T)^+ C_c$, where

$(\nabla C_c^T)^+$ denotes the pseudoinverse matrix of ∇C_c^T. Observe that if $\alpha = 0$ then $\nabla C_c^T s + \alpha C_c = 0$ contains $s = 0$ and hence for any $\sigma \in (0, 1]$, there is some $\alpha_\sigma \in (0, 1)$ for which $\{s : \nabla C_c^T s + \alpha C_c = 0\} \cap \{s : ||s|| \leq \sigma\delta_c\} \neq \emptyset$.

The drawback of the above approach is that the step depends on the parameter α, which it is not clear how to choose.

Omojokun [18], used this approach to compute a trial step that does not depend on α by choosing s_c^n to be the step that solves the following problem

$$\begin{cases} \textit{minimize} & \frac{1}{2}||\nabla C_c^T s + C_c||^2 \\ \textit{subject to} & ||s|| \leq \sigma\delta_c \end{cases}$$

for $0 < \sigma < 1$.

It might appear that Omojokun has traded the choice of α for the choice of σ, but in fact, σ is easy to choose. Some nominal value like $\sigma = 0.8$ is used throughout and the particular value of σ at a given iteration is allowed to be in some uniformly bounded strict subinterval like $(0.7, 0.9)$. This subinterval corresponds to stopping criteria on a trust-region algorithm to solve for s_c^n. See Moré [15], Moré and Sorensen [16], or Dennis and Schnabel [6].

4.2. The full-space approach. The other approach to overcoming the problem of inconsistency is the full-space approach. Algorithms based on this approach compute s_c at once in the whole $\Re^n$ space instead of considering the decomposition of the trial step. This has the advantage of avoiding the computation of a pseudoinverse solution.

The first example we know of this category of trust-region subproblems is the CDT subproblem proposed by Celis, Dennis and Tapia [4]. Instead of considering the linearized constraints $\nabla C_c^T s + C_c = 0$, they replace it by a particular inequality: $||\nabla C_c^T s + C_c|| \leq \theta_c$, where $\theta_c \in \Re$. The **CDT subproblem** can be written as follows

$$\begin{cases} \textit{minimize} & q_c(s) \\ \textit{subject to} & ||\nabla C_c^T s + C_c|| \leq \theta_c \\ & ||s|| \leq \delta_c. \end{cases}$$

The key to the CDT subproblem (and its variants) is the choice of θ_c. For more details, see Williamson [31]. Celis, Dennis, and Tapia [4] choose θ_c based on a *fraction of Cauchy decrease* condition on $||\nabla C_c^T s + C_c||$. They ask the step to satisfy, for some $r \in (0, 1]$,

$$||C_c||^2 - ||C_c + \nabla C_c^T s||^2 \geq r\{||C_c||^2 - ||\nabla C_c^T s_c^{\mathsf{cp}} + C_c||^2\}.$$

This can be done by choosing

$$\theta_c^2 = (\theta_c^{\mathsf{fcd}})^2 \equiv r||\nabla C_c^T s_c^{\mathsf{cp}} + C_c||^2 + (1 - r)||C_c||^2 \tag{4.1}$$

where $0 < r \leq 1$ and s_c^{cp} solves the problem,

$$\begin{cases} \textit{minimize} & \frac{1}{2}||\nabla C_c^T s + C_c||^2 \\ \textit{subject to} & ||s|| \leq r\delta_c \\ & s = -t\nabla C_c C_c, \qquad t \geq 0. \end{cases}$$

Note that in this case the CDT subproblem minimizes the quadratic model of ℓ over the set of steps inside the trust region that gives at least r times as much decrease in the ℓ_2-norm of the residual of the linearized constraints as does the Cauchy step.

In order to prevent the possibility of a single point for the subproblem and obtain a meaningful trust-region subproblem, it is suggested that $r < 1$, for instance $r = 0.8$.

5. A general trust-region algorithm. In this section we describe a very inclusive class of trust-region algorithms.

The typical form of trust-region algorithms for solving (EQC) is basically as follows: At the current point (x_c, λ_c), a step s_c is computed by solving some trust-region subproblems and a Lagrange multiplier vector λ_+ is obtained by using some scheme. The point (x_+, λ_+), where $x_+ = x_c + s_c$, is tested using some merit function to know whether it is a better approximation to a solution $(x_\star, \lambda_\star)$. Such merit functions often involve a parameter, which is updated using some scheme. The trust-region radius is then adjusted and a new quadratic model is formed.

In our requirements on the trust-region algorithm, the way of computing the trial steps is replaced by some conditions the steps must satisfy and the estimates of the Lagrange multiplier vector and the Hessian matrix need only be uniformly bounded. This allows the inclusion of a wide variety of trust-region algorithms and it is exactly in the spirit of Powell's Theorem 2.2 for unconstrained trust-region methods. In Section 9, we will present an example algorithm that satisfies these easy conditions.

5.1. Computing the trial steps. We first write the trial step as $s_c = s_c^t + s_c^n$, where s_c^t and s_c^n are respectively the tangential and a relaxed normal component. We do not require that s_c^n be normal to the tangent space. See Figure 1.

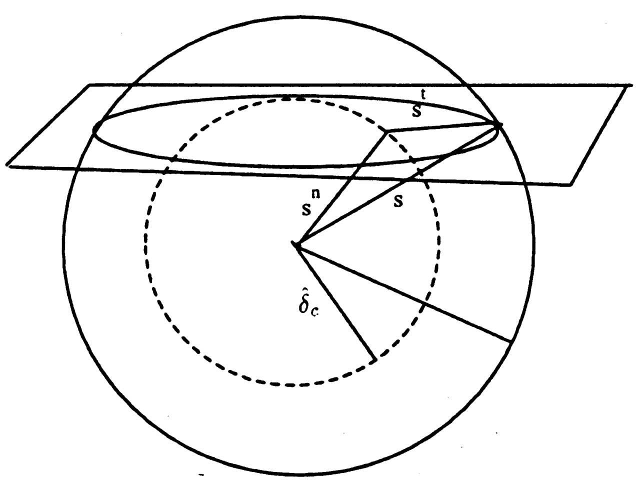

FIG. 1. *The Step* $s_c = s^n + s^t$.

We will require that the components s_c^n and s_c^t satisfy a fraction of Cauchy decrease condition on appropriate model functions. At the current iterate, if $\|C_c\| \neq 0$, then we will require that the relaxed normal component gives at least as much decrease as $-\mathsf{n}_c^{\mathsf{cp}} \nabla C_c C_c$ on the quadratic model of the linearized constraints in a trust-region of radius $r\delta_c$, where the step length $\mathsf{n}_c^{\mathsf{cp}}$ is given by

$$\mathsf{n}_c^{\mathsf{cp}} = \begin{cases} \dfrac{\|\nabla C_c C_c\|^2}{\|\nabla C_c^T \nabla C_c C_c\|^2} & \text{if } \dfrac{\|\nabla C_c C_c\|^3}{\|\nabla C_c^T \nabla C_c C_c\|^2} \leq \hat{\delta}_c \\ \dfrac{\hat{\delta}_c}{\|\nabla C_c C_c\|} & \text{otherwise,} \end{cases}$$

where $\hat{\delta}_c = r\delta_c$ and $0 < r < 1$. In words, the step s_c^n is chosen from the set of steps that satisfy a fraction of Cauchy decrease condition on the quadratic model of the linearized constraints inside $\|s\| \le \hat{\delta}_c$. Equivalently, s_c^n lies in the set

$$S_c = \{s : \|s\| \le \hat{\delta}_c\} \cap \{s : \|\nabla C_c^T s + C_c\|^2 \le (\theta_c^{\mathrm{fcd}})^2\}$$

where $(\theta_c^{\mathrm{fcd}})^2$ is given by (4.1). See Figure 2.

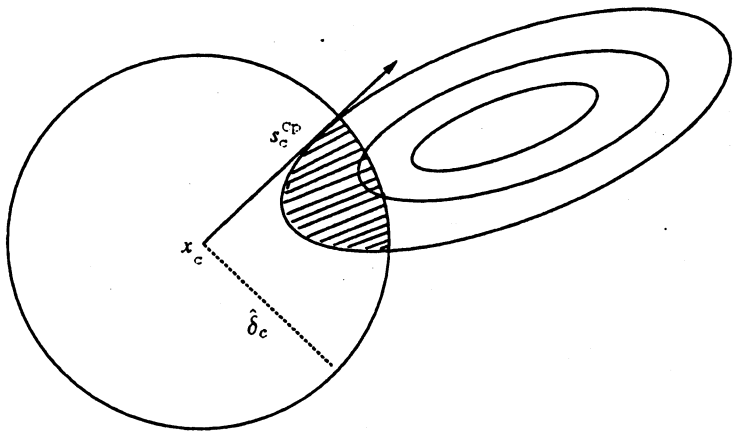

FIG. 2. *The set S_c.*

Because the relaxed normal component s_c^n is not required to be normal to the tangent space, a condition on the step is needed to ensure global convergence. In particular, the following condition is required

$$\|s_c^n\| \le K_1\|C_c\|, \tag{5.1}$$

where K_1 is some positive constant independent of the iteration.

If s_c^n is normal to the tangent space, this condition holds (see Lemma 7.1). However, this condition has to be enforced in the case when s_c^n is not normal to the tangent space. One way of enforcing this condition is to redefine $\hat{\delta}_c$ to be

$$\hat{\delta}_c = \min\{r\delta_c, K_1\|C_c\|\}, \tag{5.2}$$

where K_1 is positive constant chosen to be large enough to bound uniformly the "right inversion" operator implicitly defined by the procedure used when linear feasibility is attained. In other words, we do not suggest choosing K_1 and enforcing (5.2). Rather, we suggest (as in Section 9) that (5.1) is enforced naturally by any reasonable algorithm. It is convenient here to think of K_1 as prechosen to be greater than a uniform bound on the norm of the right inverse for $\nabla C(x)$.

We will deal with the relaxed normal components of the trial steps assuming that they satisfy (5.1).

Now we pick a linear manifold $\mathcal{M}_c$ parallel to the null-space of the constraints. Let $\mathcal{M}_c = \{s : \nabla C_c^T s = \nabla C_c^T s_c^n,\ \ s_c^n \in S_c\}$. Thus, $\mathcal{M}_c \cap \{s : \|s + s_c^n\| \le \delta_c\} \ne \emptyset$.

Observe that, in the set S_c, we are taking a fraction of δ_c, in order to forestall the case that $\mathcal{M}_c$ lies too close to the boundary of the trust-region of radius δ_c.

On the manifold $\mathcal{M}_c$, we consider a quadratic model $q_c(s)$ of the Lagrangian function associated with the (EQC) problem. Then, when $\|W_c^T \nabla q_c(s_c^n)\| \neq 0$, we ask the tangential component s_c^t to satisfy a fraction of Cauchy decrease condition from s_c^n on $q_c(s)$ reduced to $\mathcal{M}_c$. That is $s_c^n + s_c^t \in \mathcal{G}_c \cap \mathcal{M}_c$, where

$$\mathcal{G}_c = \{s_c^n + s^t : \|s^t + s_c^n\| \leq \delta_c, q_c(s_c^n + s^t) - q_c(s_c^n) \leq \sigma[q_c(s_c^n - \mathsf{t}_c^{\mathsf{cp}} W_c^T \nabla q_c(s_c^n)) - q_c(s_c^n)]\},$$

for some $0 < \sigma \leq 1$, and

$$\mathsf{t}_c^{\mathsf{cp}} = \begin{cases} \dfrac{\|W_c^T \nabla q_c(s_c^n)\|^2}{\nabla q_c(s_c^n)^T W_c \bar{H}_c W_c^T \nabla q_c(s_c^n)} & \text{if } \dfrac{\|W_c^T \nabla q_c(s_c^n)\|^3}{\nabla q_c(s_c^n)^T W_c \bar{H}_c W_c^T \nabla q_c(s_c^n)} \leq \bar{\delta}_c \\ & \text{and } \nabla q_c(s_c^n)^T W_c \bar{H}_c W_c^T \nabla q_c(s_c^n) > 0 \\ \dfrac{\bar{\delta}_c}{\|W_c^T \nabla q_c(s_c^n)\|} & \text{otherwise,} \end{cases}$$

where $\bar{H}_c$ is the reduced Hessian matrix and $\bar{\delta}_c$ is the maximum length of the step allowed inside the set $\mathcal{M}_c \cap \{s : \|s + s_c^n\| \leq \delta_c\}$ in the direction s.

It is easy to see that, $\bar{\delta}_c$ satisfies

$$(1 + r)\delta_c \; > \; \bar{\delta}_c \; > \; (1 - r)\delta_c. \tag{5.3}$$

5.2. Updating the Lagrange multiplier and the model Hessian. The method for updating the multiplier λ_c is left unspecified. We only require that the Lagrange multiplier sequence $\{\lambda_k\}$ be bounded. Any approximation to the Lagrange multiplier vector that produces a bounded sequence can be used. Also, for example, setting λ_k to a fixed vector (or even the zero vector) for all k is valid. Similarly we require only boundness of the sequence $\{H_k\}$ of approximate Hessians. Thus all $H_k = 0$ is allowed. Note that, here, we are not addressing the question of the choice of the Lagrange multiplier vector and the Hessian matrix that produces an efficient algorithm. We are addressing some weak assumptions on $\{\lambda_k\}$ and $\{H_k\}$ that produce a globally convergent algorithm. For example, our theory applies to a form of successive linear programming.

5.3. The choice of the merit function. Let x_c be the current iterate. We need to decide if a trial step chosen to satisfy $s_c^n \in S_c$ and $s_c = s_c^n + s_c^t \in \mathcal{G}_c \cap \mathcal{M}_c$ is a *good* step, that is, if the step s_c gives a new iterate x_+ that is a better approximation than x_c to a solution, say x_*, of (EQC). In constrained optimization, the meaning of better approximation should consider improvement not only in f but also in the constraint violation $\|C\|_2$. The evaluation of the trial step requires the choice of a merit function, which usually involves the objective function and the constraint violations.

Here, we use the augmented Lagrangian as a merit function

$$\mathcal{L}(x, \lambda; \rho) = f(x) + \lambda^T C(x) + \rho C(x)^T C(x), \qquad \rho > 0. \tag{5.4}$$

This function has been used as a merit function in trust-region algorithms also by Celis, Dennis, and Tapia [4], El-Alem [8], [9] and Powell and Yuan [21].

El-Alem [9] and Powell and Yuan [21] used the formula $\lambda(x) = -(\nabla C(x)^T \nabla C(x))^{-1}$ $\nabla C(x)^T C(x)$ for updating the Lagrange multiplier. For this particular choice of the multiplier, λ is a function of x and (5.4) is an exact penalty function. See Fletcher [10]. Celis, Dennis, and Tapia [4] and El-Alem [8], on the other hand, with a particular choice of the multiplier, have treated the multiplier as an independent variable.

In the context of a line search globalization strategy, Gill, Murray, Saunders, and Wright [12] and Schittkowski [22] have considered the augmented Lagrangian as a merit function. They have treated the multiplier as an independent variable and proved global convergence for their algorithms. In this work, the choice of the multiplier is left open.

Having an exact penalty function as a merit function is, of course, a desirable propriety. But it is not necessary to prove global convergence. Note that for us the merit function is used only for evaluating the steps. It is not used in choosing the trial steps.

5.4. Evaluating the trial step. Let s_c be a trial step chosen to satisfy the conditions of Section 5.1. We will accept it if sufficient improvement is produced in the merit function. To measure this improvement we compare the *actual reduction* and *predicted reduction* in the merit function from the current iterate x_c to the new one $x_+ = x_c + s_c$. The *actual reduction* is defined by

$$
\begin{aligned}
(5.5) \qquad Ared_c(s_c;\rho_c) &= \mathcal{L}(x_c,\lambda_c;\rho_c) - \mathcal{L}(x_+,\lambda_+;\rho_c) \\
&= \ell(x_c,\lambda_c) - \ell(x_+,\lambda_+) + \rho_c(\|C_c\|^2 - \|C_+\|^2),
\end{aligned}
$$

and the *predicted reduction* is defined to be

$$
(5.6) \qquad Pred_c(s_c;\rho_c) = \mathcal{L}(x_c,\lambda_c;\rho_c) - \mathcal{Q}(s_c,\Delta\lambda_c;\rho_c)
$$

where $\mathcal{Q}(s_c,\Delta\lambda_c,\rho_c) = \ell(x_c,\lambda_c) + \nabla_x\ell(x_c,\lambda_c)^T s_c + \frac{1}{2}s_c^T H_c s_c + (\Delta\lambda_c)^T(C_c + \nabla C_c^T s_c) + \rho_c(\|C_c + \nabla C_c^T s_c\|^2)$.

We will accept the step and set $x_+ = x_c + s_c$ if $\frac{Ared_c}{Pred_c} \geq \eta_1$ where $\eta_1 \in (0,1)$ is a fixed constant. A typical value for η_1 might be 10^{-4}.

5.5. Updating the trust-region radius. The strategy that we follow for updating the trust-region radius is based on the standard rules for the unconstrained case. More details can be found in Dennis and Schnabel [6] or Fletcher [11]. However for our global convergence theory, a modification in the strategy of updating the trust-region radius is introduced. The reader will see that this modification is of no importance in practice; it is merely a formality. At the beginning we set constants $\delta_{\max} \geq \delta_{\min}$ and each time we find an acceptable step, we start the next iteration with a value of $\delta_+ \geq \delta_{\min}$. In short, δ_c can be reduced below $\delta_{\min}$ while seeking an acceptable step, but $\delta_+ \geq \delta_{\min}$ must hold at the beginning of the next iteration after finding an acceptable step. The following is the scheme for evaluating the step and updating the trust-region radius.

ALGORITHM 5.1. **Evaluating the step and updating the trust-region radius**

Given the constants: $0 < \alpha_1 < 1$, $\alpha_2 > 1$ *and* $0 < \eta_1 < \eta_2 < 1$ *and* $\delta_{\max} \geq \delta_c \geq \delta_{\min} > 0$.

While $\frac{Ared_c}{Pred_c} < \eta_1$ (* *e.g.* $\eta_1 = 10^{-4}$ *)
- *Do not accept the step.*
- *Reduce the trust-region radius:* $\delta_c \leftarrow \alpha_1\|s_c\|$ (* *e.g.* $\alpha_1 = 0.5$ *), *and compute a new trial step* s_c.

End while

If $\eta_1 \leq \frac{Ared_c}{Pred_c} < \eta_2$ (* *e.g.* $\eta_2 = 0.5$ *) **then**
- *Accept the step:* $x_+ = x_c + s_c$.
- *Set the trust-region radius:* $\delta_+ = \max\{\delta_c, \delta_{\min}\}$.

End if

If $\frac{Ared_c}{Pred_c} \geq \eta_2$ **then**
Accept the step: $x_+ = x_c + s_c$.
Increase the trust-region radius:

$$\delta_+ = \min\{\delta_{\max}, \max\{\delta_{\min}, \alpha_2\delta_c\}\} \tag{5.7}$$

(e.g.* $\alpha_2 = 2$ **).*
End if

It is worth noting that in practice one might have another branch in which some $\eta_{\frac{3}{2}} \in (\eta_1, \eta_2)$ is used to reduce the trust-region radius if $\eta_1 \leq \frac{Ared_c}{Pred_c} \leq \eta_{\frac{3}{2}}$. A typical value for $\eta_{\frac{3}{2}}$ is .1, and the motivation is to try to avoid the expense of a next unacceptable trial step. Of course, in our work we would have to decrease the trust-region radius only when the step is rejected. Another modification sometimes used in practice is to allow internal doubling. This can be viewed loosely as letting α_2 in (5.7) depend on $\frac{Ared_c}{Pred_c}$. See Dennis and Schnabel, Page 144, [6].

Observe that in (5.5) and (5.6) we have expressed the quantities *Ared* and *Pred* as functions of ρ. Thus ρ_c does not effect the choice of the trial step s_c but we need to determine ρ_c *before* evaluating the step s_c. The right choice of the penalty parameter is one of the most important issues for algorithms that use the augmented Lagrangian as a merit function. This takes us to the following section.

5.6. The penalty parameter. Numerical experience with nonlinear programming algorithms that use the augmented Lagrangian as a merit function has shown that good performance of the algorithm depends on keeping the penalty parameter as small as possible. See Gill, Murray, Saunders and Wright [13]. On the other hand, global convergence theories developed by El-Alem [7], [8] and Powell and Yuan [21], require that the sequence $\{\rho_k\}$ be nondecreasing. El-Alem [7] requires that ρ be chosen so that the predicted decrease in the merit function be at least as much as the decrease in $||\nabla C_c^T s + C_c||^2$.

We consider, as an update formula for the penalty parameter, El-Alem's scheme given in [8], since it ensures that the merit function is decreased at each iteration by at least a fraction of Cauchy decrease in the quadratic model of the constraints. This indicates compatibility with the fraction of Cauchy decrease conditions imposed on the trial steps. In addition, good performance was reported when implementing this scheme. See Williamson [31]. It can be stated as follows

ALGORITHM 5.2. **Updating the penalty parameter**

1. **Initialization**
   Set $\rho_0 = 1$ *and choose a small constant* $\beta > 0$.
2. **At the current iterate x_c, after s_c has been chosen:**
   Compute

$$Pred_c(s_c;\rho_-) = q_c(0) - q_c(s_c) - \Delta\lambda_c^T(C_c + \nabla C_c^T s) + \rho_-[||C_c||^2 - ||\nabla C_c^T s_c + C_c||^2].$$

If $Pred_c(s_c;\rho_-) \geq \frac{\rho_-}{2}[||C_c||^2 - ||\nabla C_c^T s_c + C_c||^2]$,
then *set* $\rho_c = \rho_-$,
else *set* $\rho_c = \bar{\rho}_c + \beta$, *where*

$$\bar{\rho}_c = \frac{2[q_c(s_c) - q_c(0) + \Delta\lambda_c^T(C_c + \nabla C_c^T s)]}{||C_c||^2 - ||\nabla C_c^T s_c + C_c||^2}.$$

End if

The initial choice of the penalty parameter ρ_0 is arbitrary and scale dependent. Here, we take $\rho_0 = 1$ for convenience.

An immediate consequence of the above algorithm is that, at the current iteration, we have

$$Pred_c(s_c;\rho_c) \geq \frac{\rho_c}{2}[\|C_c\|^2 - \|C_c + \nabla C_c^T s_c\|^2]. \tag{5.8}$$

5.7. Termination of the algorithm. We use first order necessary conditions for problem (EQC) to terminate the algorithm. The algorithm is terminated if $\|W_c^T\nabla_x \ell_c\| + \|C_c\| \leq \varepsilon_{tol}$ where $\varepsilon_{tol} > 0$ is a pre-specified constant and W_c is a matrix with columns forming a basis for the null space. We require that $\{W_k\}$ be uniformly bounded in norm for all k.

6. Statement of the algorithm.

We present a formal description of our class of nonlinear programming algorithms.

ALGORITHM 6.1. **The NLP-algorithm.**

step 0. *(Initialization)*
Given x_0, λ_0, *compute* W_0.
Choose $\delta_{\min}$, $\delta_{\max}$, *and* $\varepsilon_{tol} > 0$.
Set $\rho_0 = 1$ *and* $\beta > 0$.

step 1. *(Test for convergence)*
If $\|W_c^T\nabla_c\ell(x_c)\| + \|C(x_c)\| \leq \varepsilon_{tol}$
then *terminate.*
End if

step 2. *(Compute a trial step)*
If x_c *is feasible* **then**
a) *find a step* s_c^t *that satisfies a fraction of Cauchy decrease condition on the quadratic model* $q_c(s)$ *of the Lagrangian around* x_c. *(See Section 5.1)*
b) *Set* $s_c = s_c^t$.
else (* $C(x_c) \neq 0$ *)
a) *Compute* s_c^n *that satisfies a fraction of Cauchy decrease condition on the quadratic model of the linearized constraints. (See Section 5.1)*
b) **If** $\|W_c^T\nabla q(s_c^n)\| = 0$
then *set* $s_c^t = 0$
else *find* s_c^t *that satisfies a fraction of Cauchy decrease condition on the quadratic model* $q_c(s_c^n + s)$ *from* s_c^n. *(See Section 5.1)*
End if
c) *Set* $s_c = s_c^n + s_c^t$.
End if

step 3. *(Update* λ_c*)*
Choose a Lagrange multiplier vector λ_+.
Set $\Delta\lambda_c = \lambda_+ - \lambda_c$.

step 4. *(Update the penalty parameter)*
Compute

$$\begin{aligned} Pred_c(s_c;\rho_-) &= q_c(0) - q_c(s_c) - \Delta\lambda_c^T(C_c + \nabla C_c^T s) \\ &\quad + \rho_-[\|C_c\|^2 - \|\nabla C_c^T s_c + C_c\|^2]. \end{aligned}$$

Update ρ_- *to obtain* ρ_c *by using* **Algorithm 5.2.**

step 5. *(Evaluate the step)*
Compute

$$Ared_c(s_c;\rho_c) \;=\; \ell(x_c,\lambda_c) - \ell(x_+,\lambda_+) + \rho_c(\|C_c\|^2 - \|C_+\|^2).$$

Evaluate the step and update the trust region radius by using **Algorithm 5.1.**
If *the step is accepted*
then *update H_c and go to* **step 1.**
else
go to **step 2.**
End if

The above represents a typical trust-region algorithm for solving problem (EQC). We leave the way of computing the trial steps undefined. This will allow the inclusion of a wide variety of trial step calculation techniques. For similar reasons we left the way of updating the Lagrange multiplier vector and the Hessian matrix undefined.

In the next two sections we prove global convergence of the above algorithm class.

7. The global convergence theory. Before beginning our global convergence theory, let us give an overview of the steps that comprise this theory.

The trial step is shown to satisfy a sufficient predicted decrease condition, the fraction of Cauchy decrease. Note that in our algorithm, we assume that the tangential and the relaxed normal components of the trial step each satisfy this condition. In Lemma 7.1, we will express this in a technical form similar to inequality (2.2).

The definition of predicted reduction is shown to give an approximation to the actual reduction that is accurate to within the square of the step length. This is proved in Lemma 7.5.

The penalty parameter ρ_k is shown to be bounded. The technique is to prove that, at any iteration k at which the penalty parameter is increased, the product of the penalty parameter and the trust-region radius is bounded by a constant that does not depend on k, this is done in Lemma 7.10, and the sequence of the trust-region radii is shown to be bounded away from zero. This is proved in Lemma 7.11. The proof of this lemma shows the crucial role that is played by setting the trust region to be no smaller than $\delta_{\min}$ after every acceptable step. See Section 5.5. Finally, the penalty parameter ρ_k is shown to be bounded. The proof is given in Lemma 7.12.

The algorithm is shown to be well-defined in the sense that it always finds an acceptable step after finitely many unacceptable trials. This result is proved in Theorem 8.1. Using the above results and Theorem 8.1, the trust-region radius is shown to be bounded away from zero. The proof is given in Lemma 8.2.

Finally, In Theorem 8.3, it is shown that the algorithm always terminates. *i. e.*, the termination condition of the algorithm will be met after finitely many iterations.

Now we start by stating the assumptions under which global convergence is proved. The same assumptions as A1 - A5 (see below) are used by Byrd, Schnabel, and Shultz [2], El-Alem [7], [8], [9] and Powell and Yuan [21] and their particular choices of Lagrange multiplier vectors satisfy A6.

7.1. The standard assumptions. We state assumptions needed to prove global convergence for the nonlinear programming Algorithm 6.1.

Let the sequence of iterates $\{x_k\}$ generated by the algorithm satisfy:

A1. For all k, x_k and $x_k + s_k \in \Omega$, where Ω is a convex set of $\Re^n$.

A2. $f, C \in C^2(\Omega)$.

A3. $rank(\nabla C(x)) = m$ for all $x \in \Omega$.

A4. $f(x), \nabla f(x)$, $\nabla^2 f(x)$, $C(x)$, $\nabla C(x)$, $(\nabla C(x)^T \nabla C(x))^{-1}$, $W(x)$, and $\nabla^2 C_i(x)$ for $i = 1, \cdots, m$ are all uniformly bounded in Ω.

A5. The matrices $H_k, k = 1, 2, ..$ are all uniformly bounded in Ω.

A6. The Lagrange multiplier vectors $\lambda_k, k = 1, 2, ..$ are all uniformly bounded in Ω.

Assumption A4 means that for all $x \in \Omega$, there exist positive constants ν, ν_0, ν_1, ν_2, ν_3, ν_4, ν_5, and ν_6 such that:
$\|f(x)\| \le \nu$, $\|\nabla f(x)\| \le \nu_0$, $\|C(x)\| \le \nu_1$, $\|\nabla C(x)\| \le \nu_2$, $\|(\nabla C(x)^T \nabla C(x))^{-1}\| \le \nu_3$, $\|\nabla^2 f(x)\| \le \nu_4$, $\|\nabla^2 c_i(x)\| \le \nu_5 \ \ \forall\, i = 1, \cdots, m$, and $\|W(x)\| \le \nu_6$.

An immediate consequence of Assumptions A4 and A5 is the existence of a constant $\nu_7 > 0$ that does not depend on k such that $\|H_k\| \le \nu_7$, $\|W_k^T H_k\| \le \nu_7$, and $\|W_k^T H_k W_k\| \le \nu_7$.

Assumption A6 means that for all $x \in \Omega$, there exists a constant $\nu_8 > 0$ that does not depend on k, such that $\|\lambda_k\| \le \nu_8$.

The following three subsections are devoted to presenting lemmas needed to prove global convergence.

7.2. Properties of the trial step. The following lemma shows that (5.1) holds for the component s_c^n of s_c when it is normal to the tangent space. It is an immediate consequence of the standard assumptions.

LEMMA 7.1. *At the current iterate x_c, let the component s_c^n be normal to the tangent space, then under the standard assumptions, there exists a constant $K_1 > 0$ independent of the iterates, such that*

$$\|s_c^n\| \ \le \ K_1 \|C_c\|. \tag{7.1}$$

Proof. The proof follows directly from the standard assumptions and the fact that $\|C_k + \nabla C_k^T s_k\| \le \|C_k\|$. □

The following lemma expresses in a workable form the pair of fraction of Cauchy decrease conditions imposed on the trial steps.

LEMMA 7.2. *Let the trial steps satisfy the conditions given in step 2 of Algorithm 6.1, then under the standard assumptions there exist constants K_2, K_3, and K_4 independent of the iterates such that*

$$\|C_k\|^2 - \|C_k + \nabla C_k^T s_k\|^2 \ge K_2 \|C_k\| \min\{K_3 \|C_k\|, r\delta_k\}, \tag{7.2}$$

and

$$q_k(s_k^n) - q_k(s_k) \ge \frac{1}{2} \|W_k^T \nabla q_k(s_k^n)\| \min\{(1 - r)\delta_k, K_4 \|W_k^T \nabla q_k(s_k^n)\|\}. \tag{7.3}$$

Proof. The proof is an application of Lemma 2.1 to the two subproblems, followed by a use of the standard assumptions and (5.3). □

Now we deal with the trial steps assuming that they satisfy inequalities (7.2) and (7.3).

LEMMA 7.3. *Under the standard assumptions, there exists a constant $K_5 > 0$ independent of the iterates, such that*

$$q_k(0) - q_k(s_k^n) - \Delta\lambda_k^T (C_k + \nabla C_k^T s_k) \ge -K_5 \|C_k\|. \tag{7.4}$$

Proof. Consider

$$\begin{aligned} q_k(0) - q_k(s_k^n) &= -\nabla_x \ell_k^T s_k^n - \frac{1}{2}(s_k^n)^T H_k s_k^n \\ &\geq -\|\nabla_x \ell_k\| \, \|s_k^n\| - \frac{1}{2}\|H_k\| \, \|s_k^n\|^2 \\ &= -(\|\nabla_x \ell_k\| + \frac{1}{2}\|H_k\| \, \|s_k^n\|) \, \|s_k^n\|. \end{aligned}$$

Using (5.1), the fact that $\|s_k^n\| < \delta_{\max}$, λ_k and $\Delta\lambda_k$ are bounded, $\|C_k + \nabla C_k^T s_k\| \leq \|C_k\|$, and the standard assumptions, we have

$$q_k(0) - q_k(s_k^n) - \Delta\lambda_k^T (C_k + \nabla C_k^T s_k) \quad \geq \quad -K_5\|C_k\|,$$

and we obtain the desired result. □

The following lemma gives an upper bound on the difference between the actual reduction and the predicted reduction.

LEMMA 7.4. *Under the standard assumptions, there exist positive constants K_6, K_7 and K_8, independent of k, such that*

$$|Ared_k(s_k;\rho_k) - Pred_k(s_k;\rho_k)| \leq K_6\|s_k\|^2 + K_7\rho_k\|s_k\|^3 + K_8\rho_k\|s_k\|^2\|C_k\|. \tag{7.5}$$

Proof. The proof follows directly from El-Alem [8]. □

If the penalty parameter were uniformly bounded, the next lemma would show that the predicted reduction provides an approximation to the actual merit function's reduction that is accurate to within the square of the step length.

LEMMA 7.5. *Under the standard assumptions, there exists a constant $K_9 > 0$ that does not depend on k, such that*

$$|Ared_k(s_k;\rho_k) - Pred_k(s_k;\rho_k)| \leq K_9\rho_k\|s_k\|^2. \tag{7.6}$$

Proof. The proof follows directly from the above lemma, the fact that $\|s_k\|$ and $\|C_k\|$ are bounded, and $\rho_k \geq 1$. □

7.3. The decrease in the model. This section deals with the decrease in the merit function by the trial step. We start with the following lemma.

LEMMA 7.6. *Let s_c be generated by Algorithm 6.1. Then under the standard assumptions, the predicted decrease in the merit function satisfies*

$$\begin{aligned} Pred_c(s_c;\rho_c) \quad \geq \quad & \frac{1}{2}\|W_c^T \nabla q_c(s_c^n)\| \min\{K_4\|W_c^T \nabla q_c(s_c^n)\| \, , \, (1-r)\delta_c\} \\ & - K_5\|C_c\| + \rho_c[\|C_c\|^2 - \|\nabla C_c^T s_c + C_c\|^2], \end{aligned} \tag{7.7}$$

where K_5 is as in Lemma 7.3.

Proof. We have

$$\begin{aligned} Pred_c(s_c;\rho_c) &= q_c(0) - q_c(s_c) - \Delta\lambda_c^T(C_c + \nabla C_c^T s_c) \\ &\quad + \rho_c[\|C_c\|^2 - \|\nabla C_c^T s_c + C_c\|^2] \\ &= (q_c(s_c^n) - q_c(s_c)) \\ &\quad + (q_c(0) - q_c(s_c^n)) - \Delta\lambda_c^T(C_c + \nabla C_c^T s_c) \\ &\quad + \rho_c[\|C_c\|^2 - \|\nabla C_c^T s_c + C_c\|^2]. \end{aligned}$$

From (7.3) and Lemma 7.3, we have

$$\begin{aligned} Pred_c(s_c;\rho_c) \;&\geq\; \frac{1}{2}\|W_c^T\nabla q_c(s_c^n)\| \min\{K_4\|W_c^T\nabla q_c(s_c^n)\|\,,\,(1-r)\delta_c\} \\ &\quad - K_5\|C_c\| + \rho_c[\|C_c\|^2 - \|\nabla C_c^T s_c + C_c\|^2]. \end{aligned}$$

Hence the result is established. □

If x_c is feasible, then the predicted reduction does not depend on ρ_c, so we take ρ_c as the penalty parameter from the previous iteration. The question now is how near to feasibility must an iterate be in order that the penalty parameter need not be increased. The answer is given by the following lemma.

LEMMA 7.7. *Assume that the algorithm does not terminate at the current iterate. If $\|C_c\| \leq \alpha\delta_c$ where α is a constant that satisfies:*

$$\alpha \leq \min\left\{\frac{\varepsilon_{tol}}{3\delta_{\max}}\,,\,\frac{\varepsilon_{tol}}{3\nu_7 K_1\delta_{\max}}\,,\,\frac{\varepsilon_{tol}}{12K_5}\min\{\frac{K_4\varepsilon_{tol}}{3\delta_{\max}}\,,\,1-r\}\right\} \tag{7.8}$$

then

$$\begin{aligned} Pred_c(s_c;\rho_-) \;&\geq\; \frac{1}{4}\|W_c^T\nabla q_c(s_c^n)\| \min\{K_4\|W_c^T\nabla q_c(s_c^n)\|\,,\,(1-r)\delta_c\} \\ &\quad + \rho_- K_2\|C_c\| \min\{K_3\|C_c\|, r\delta_c\}. \end{aligned} \tag{7.9}$$

Proof. If the algorithm does not terminate at x_c, then $\|W_c^T\nabla_x\ell_c\| + \|C_c\| > \varepsilon_{tol}$, and since $\|C_c\| \leq \alpha\delta_c$ with $\alpha \leq \frac{\varepsilon_{tol}}{3\delta_{\max}}$, therefore, $\|C_c\| \leq \frac{\varepsilon_{tol}}{3}$ and the reduced gradient satisfies $\|W_c^T\nabla_x\ell_c\| > \frac{2}{3}\varepsilon_{tol}$. Now,

$$\begin{aligned} \|W_c^T\nabla q_c(s_c^n)\| \;&=\; \|W_c^T(\nabla_x\ell_c + H_c s_c^n)\| \geq \|W_c^T\nabla_x\ell_c\| - \|W_c^T H_c s_c^n\| \\ &\geq\; \frac{2}{3}\varepsilon_{tol} - \nu_7 K_1\|C_c\| \geq \frac{2}{3}\varepsilon_{tol} - \nu_7 K_1\alpha\delta_c. \end{aligned}$$

But since $\alpha \leq \frac{\varepsilon_{tol}}{3\nu_7 K_1\delta_{\max}}$, it follows that

$$\|W_c^T\nabla q_c(s_c^n)\| \geq \frac{1}{3}\varepsilon_{tol}.$$

From Lemma 7.6, we have

$$\begin{aligned} Pred_c(s_c;\rho_-) \;&\geq\; \frac{1}{2}\|W_c^T\nabla q_c(s_c^n)\| \min\{(1-r)\delta_c\,,\,K_4\|W_c^T\nabla q_c(s_c^n)\|\} \\ &\quad - K_5\|C_c\| + \rho_-[\|C_c\|^2 - \|\nabla C_c^T s_c + C_c\|^2]. \end{aligned}$$

Since $\|W^T\nabla q(s_c^n)\| > \frac{1}{3}\varepsilon_{tol}$, we have

$$\begin{aligned} Pred_c(s_c;\rho_-) \;&\geq\; \frac{1}{4}\|W_c^T\nabla q_c(s_c^n)\| \min\{(1-r)\delta_c\,,\,K_4\|W_c^T\nabla q_c(s_c^n)\|\} \\ &\quad + \frac{1}{12}\varepsilon_{tol}\min\left\{(1-r)\delta_c\,,\,\frac{\varepsilon_{tol}K_4}{3}\right\} \\ &\quad - K_5\alpha\delta_c + \rho_-[\|C_c\|^2 - \|\nabla C_c^T s_c + C_c\|^2]. \end{aligned}$$

Thus

$$\begin{aligned} Pred_c(s_c;\rho_-) \;\geq\; & \frac{1}{4}\|W_c^T\nabla q_c(s_c^n)\| \min\{(1-r)\delta_c\,,\; K_4\|W_c^T\nabla q_c(s_c^n)\|\} \\ & + \frac{\varepsilon_{tol}\delta_c}{12}\min\left\{1-r\,,\; \frac{\varepsilon_{tol}K_4}{3\delta_{\max}}\right\} \\ & - K_5\alpha\delta_c + \rho_-[\|C_c\|^2 - \|\nabla C_c^T s_c + C_c\|^2], \end{aligned}$$

and since

$$\alpha \leq \frac{\varepsilon_{tol}}{12K_5}\min\left\{\frac{K_4\varepsilon_{tol}}{3\delta_{\max}}\,,\; 1-r\right\},$$

we have

$$\begin{aligned} Pred_c(s_c;\rho_-) \;\geq\; & \frac{1}{4}\|W_c^T\nabla q_c(s_c^n)\| \min\{K_4\,\|W_c^T\nabla q_c(s_c^n)\|\,,\; (1-r)\delta_c\} \\ & + \rho_-[\|C_c\|^2 - \|\nabla C_c^T s_c + C_c\|^2]. \end{aligned} \tag{7.10}$$

Now, using (7.2), we obtain the desired result. □

Inequality (7.10) guarantees that if the algorithm does not terminate and if $\|C_c\| \leq \alpha\delta_c$, then the penalty parameter at the current iteration does not need to be increased in *step 2* of Algorithm 6.1. This is equivalent to saying that the possible increases in the penalty parameter will be only when $\|C_c\| > \alpha\delta_c$.

LEMMA 7.8. *There exists a constant $K_{10} > 0$, such that at any iteration at which the algorithm does not terminate and $\|C_k\| \leq \alpha\delta_k$ where α is as in Lemma 7.7, the following inequality holds*

$$Pred_k(s_k;\rho_k) \;\geq\; K_{10}\delta_k. \tag{7.11}$$

Proof. Since the algorithm does not terminate and $\|C_k\| \leq \alpha\delta_k$, where α is as in (7.8), then from (7.9) and using a similar argument as in Lemma 7.7, we can write

$$Pred_k(s_k;\rho_k) \geq \frac{\varepsilon_{tol}}{12}\min\left\{(1-r)\delta_k,\; \frac{K_4\varepsilon_{tol}}{3}\right\} \geq \frac{\varepsilon_{tol}}{12}\min\left\{(1-r),\; \frac{K_4\varepsilon_{tol}}{3\delta_{\max}}\right\}\delta_k.$$

Defining

$$K_{10} = \frac{\varepsilon_{tol}}{12}\min\left\{1-r,\; \frac{K_4\varepsilon_{tol}}{3\delta_{\max}}\right\},$$

we have $Pred_k(s_k;\rho_k) \geq K_{10}\delta_k$ and this is the desired result. □

In the next section we will discuss the role of the penalty parameter in the global convergence of the nonlinear programming algorithm.

7.4. The behavior of the penalty parameter. In this section we discuss the behavior of the penalty parameter. The crucial result here is that the sequence of trust-region radii $\{\delta_k\}$ is bounded away from zero. This will allow us to conclude that the sequence $\{\rho_k\}$ of penalty parameters is bounded.

According to the rule for updating the penalty parameter, we use the penalty parameter from the previous iteration if the amount of predicted decrease with the old penalty parameter is at least a fraction of the decrease in the quadratic model of the linearized constraints, that is, if

$$Pred_c(s_c;\rho_-) \geq \frac{\rho_-}{2}[\|C_c\|^2 - \|C_c + \nabla C_c^T s_c\|^2], \tag{7.12}$$

then $\rho_c = \rho_-$. Otherwise, we use $\rho_c = \bar{\rho}_c + \beta$, which enforces (5.8). See Section 5.6.

LEMMA 7.9. *Let $\{\rho_k\}$ be the sequence of penalty parameters generated by the algorithm, then*

1. *$\{\rho_k\}$ forms a nondecreasing sequence.*
2. *If the penalty parameter is increased, it will increase by at least β.*
3. *If the penalty parameter is not increased, then inequality (7.12) will also hold.*

Proof. The proof is straightforward. □

LEMMA 7.10. *If ρ_k is increased at the current iterate, then there exists a constant $K_{11} > 0$ that does not depend on k, such that*

$$\rho_k \delta_k \leq K_{11}. \tag{7.13}$$

Proof. If ρ_k is increased at the current iterate, then it is updated by the rule

$$\rho_k = \frac{2[q_k(s_k) - q_k(0) + \Delta\lambda_k^T(C_k + \nabla C_k^T s_k)]}{\|C_k\|^2 - \|C_k + \nabla C_k^T s_k\|^2} + \beta.$$

Hence,

$$\begin{aligned}
\frac{\rho_k}{2}[\|C_k\|^2 - \|C_k + \nabla C_k^T s_k\|^2] &= [q_k(s_k) - q_k(0)] + \Delta\lambda_k^T(C_k + \nabla C_k^T s_k) \\
&\quad + \frac{\beta}{2}[\|C_k\|^2 - \|C_k + \nabla C_k^T s_k\|^2] \\
&= [q_k(s_k) - q_k(s_k^n)] \\
&\quad + [q_k(s_k^n) - q_k(0)] + \Delta\lambda_k^T(C_k + \nabla C_k^T s_k) \\
&\quad + \frac{\beta}{2}[-2(\nabla C_k C_k)^T s_k - \|\nabla C_k^T s_k\|^2].
\end{aligned}$$

Applying (7.2) to the left-hand side, and (7.3) and Lemma 7.3 to the right-hand side, we can obtain the following:

$$\begin{aligned}
\frac{\rho_k K_2}{2}\|C_k\| \min\{ r\delta_k \;,\; K_3\|C_k\| \} \\
&\leq -\frac{1}{2}\|W_k^T \nabla q_k(s_k^n)\| \min\{K_4\|W_k^T\nabla q_k(s_k^n)\| \,,\, (1-r)\delta_k\} \\
&\quad + K_5\|C_k\| - \beta(\nabla C_k C_k)^T s_k - \frac{\beta}{2}\|\nabla C_k^T s_k\|^2 \\
&\leq K_5\|C_k\| - \beta(\nabla C_k C_k)^T s_k \\
&\leq K_5\|C_k\| + \beta\|\nabla C_k\|\,\|C_k\|\,\|s_k\| \\
&\leq (K_5 + \beta\|\nabla C_k\|\,\|s_k\|)\|C_k\|.
\end{aligned}$$

Then,

$$\rho_k \frac{K_2}{2} \min\{r\delta_k \,,\, K_3\|C_k\|\} \leq K_5 + \beta\nu_2\delta_{\max}.$$

Since at the current iterate the penalty parameter increases, then from Lemma 7.7 we have $||C_k|| > \alpha\delta_k$. Hence

$$\rho_k \frac{K_2}{2} \min \{r\delta_k \ , \ K_3\alpha\delta_k\} \leq K_5 + \beta\nu_2\delta_{\max}$$

and

$$\rho_k\delta_k \leq \frac{2K_5 + 2\beta\nu_2\delta_{\max}}{K_2 \min \{r \ , \ K_3\alpha\}}.$$

Defining

$$K_{11} = \frac{2K_5 + 2\beta\nu_2\delta_{\max}}{K_2 \min \{r \ , \ K_3\alpha\}},$$

we obtain the desired result. $\square$

The following lemma gives a lower bound for the sequence $\{\delta_k\}$ for those iterates at which the algorithm does not terminate and the penalty parameter is increased.

LEMMA 7.11. *Under the standard assumptions, there exists a constant $\tilde{\delta}$, which does not depend on the iterates, such that for any k at which the algorithm does not terminate and the penalty parameter is increased,*

$$\delta_k \ \geq \ \tilde{\delta}. \tag{7.14}$$

Proof. Let s_k be the current step (this step can be acceptable or unacceptable) and let s_{k-1} be the last acceptable step. Let us denote the indices of unacceptable steps between s_{k-1} and s_k by $k_1, k_2, \ldots, k_j$. That is

$$s_{k-1}, \underbrace{s_{k_1}, s_{k_2}, \cdots, s_{k_j}}_{\text{rejected}}, s_{k_{j+1}} = s_k.$$

We consider three cases:

i) $s_{k_1} = s_k$, that is, there are no unacceptable steps between s_{k-1} and s_k.

ii) $s_{k_1} \neq s_k$ and $||C_k|| > \alpha\delta_{k_i}$ for all $i = 1, \ldots, j+1$.

iii) $s_{k_1} \neq s_k$ but $||C_k|| > \alpha\delta_{k_i}$ does not hold for all $i = 1, \ldots, j+1$.

i) If $s_k = s_{k_1}$, then from the way of updating the trust-region radius,

$$\delta_k \ \geq \ \max\{\delta_{k-1}, \delta_{\min}\} \ \geq \ \delta_{\min}. \tag{7.15}$$

ii) If $s_{k_1} \neq s_k$ and at the same time the constraint violation $||C_k|| > \alpha\delta_{k_i}$ for all $i = 1, \cdots, j$, then from Lemma 7.5, we have

$$|Ared_k(s_{k_i}; \rho_{k_i}) - Pred_k(s_{k_i}; \rho_{k_i})| \leq K_9\rho_{k_i}||s_{k_i}||^2.$$

Now since $||C_k|| > \alpha\delta_{k_i}$, then from the way of updating ρ_{k_i} and using inequality (7.2), we have

$$\begin{aligned} Pred_k(s_{k_i}; \rho_{k_i}) \ &\geq \ \frac{\rho_{k_i}}{2}[||C_k||^2 - ||C_k + \nabla C_k^T s_{k_i}||^2] \\ &\geq \ \frac{\rho_{k_i}}{2} K_2||C_k|| \min\{K_3\alpha, \ r\}\delta_{k_i}. \end{aligned}$$

Hence

$$(7.16)\qquad \frac{|Ared_k(s_{k_i};\rho_{k_i}) - Pred_k(s_{k_i};\rho_{k_i})|}{Pred_k(s_{k_i};\rho_{k_i})} \le \frac{2K_9\|s_{k_i}\|}{K_2\|C_k\|\min\{K_3\alpha,\, r\}}.$$

Since all the steps s_{k_i} for $i = 1, \cdots, j$ are rejected, it must be the case that

$$(7.17)\qquad (1-\eta_1) < \left| \frac{Ared_k(s_{k_i};\rho_{k_i})}{Pred_k(s_{k_i};\rho_{k_i})} - 1 \right|.$$

So from (7.16) and (7.17), we have

$$(7.18)\qquad \|s_{k_i}\| \ge \frac{(1-\eta_1)K_2\min\{\alpha K_3,\, r\}}{2K_9}\|C_k\|, \quad \forall i = 1, \cdots, j.$$

Since $\delta_{k_{j+1}} \ge \alpha_1\|s_{k_j}\|$, and since $\|C_k\| > \alpha\delta_{k_1}$, it follows that

$$(7.19)\qquad \delta_k \ge \alpha_1\|s_{k_j}\| \ge \alpha_1 \left[\frac{(1-\eta_1)K_2\min\{\alpha K_3,\, r\}}{2K_9}\right]\alpha\delta_{k_1}.$$

Now, according to the rule for updating the trust-region radius,

$$\delta_{k_1} = \max\{\delta_c, \delta_{\min}\} \ge \delta_{\min}.$$

Then

$$\delta_k \ge \frac{\alpha_1(1-\eta_1)K_2\min\{\alpha K_3,\, r\}}{2K_9}\alpha\delta_{\min} = K_{12}.$$

iii) If $s_{k_1} \ne s_k$ and $\|C_k\| > \alpha\delta_{k_i}$ does not hold for all $i = 1, \cdots, j+1$, then there exists at least one index p such that $\|C_k\| \le \alpha\delta_{k_p}$. Let l be the largest index such that $\|C_k\| \le \alpha\delta_{k_l}$ holds.

$$s_{k-1}, \overbrace{\underbrace{s_{k_1}, s_{k_2}, \cdots, s_{k_l}}_{\|C_k\|\le\alpha\delta_{k_i}}, \underbrace{s_{k_{l+1}}, \cdots, s_{k_j}}_{\|C_k\|>\alpha\delta_{k_i}}}^{\text{rejected}}, s_{k_{j+1}} = s_k.$$

Observe that for all indices i such that $l+1 \le i \le j+1$, we have $\|C_k\| > \alpha\delta_{k_i}$, then as in the first two cases we will see that there exists a constant K_{13} such that

$$(7.20)\qquad \delta_k \ge K_{13}\|s_{k_l}\|,$$

If $s_{k_{l+1}} = s_k$, then

$$(7.21)\qquad \delta_k \ge \alpha_1\|s_{k_l}\|.$$

If $s_{k_{l+1}} \ne s_k$, since $\|C_k\| > \alpha\delta_{k_i}$ for all $i = l+1, \ldots, j+1$ from (7.19), we have

$$(7.22)\qquad \delta_k \ge \alpha_1 \left[\frac{(1-\eta_1)K_2\min\{\alpha K_3,\, r\}}{2K_9}\right]\alpha\|s_{k_l}\|.$$

Defining

$$K_{13} = \min\left\{\alpha_1, \alpha_1\left[\frac{(1-\eta_1)K_2\min\{\alpha K_3,\, r\}}{2K_9}\right]\alpha\right\}$$

we obtain (7.20). Now, since $\|C_k\| \leq \alpha\delta_{k_l}$, the penalty parameter does not need to be increased. From (7.5) we have

$$|Ared_k(s_{k_l};\rho_{k_l}) - Pred_k(s_{k_l};\rho_{k_l})| \leq [K_6 + (K_7 + \alpha K_8)\rho_{k_l}\|s_{k_l}\|]\|s_{k_l}\|\delta_{k_l}.$$

But

$$\rho_{k_l}\|s_{k_l}\| \leq \rho_k \frac{\delta_k}{K_{13}} \leq \frac{K_{11}}{K_{13}} = K_{14}.$$

Therefore,

$$|Ared_k(s_{k_l};\rho_{k_l}) - Pred_k(s_{k_l};\rho_{k_l})| \leq [K_6 + (K_7 + \alpha K_8)K_{14}]\|s_{k_l}\|\delta_{k_l}. \tag{7.23}$$

Also, since $\|C_k\| \leq \alpha\delta_{k_l}$, then from Lemma 7.8, we have

$$Pred_k(s_{k_l};\rho_{k_l}) \geq K_{10}\delta_{k_l}. \tag{7.24}$$

Using (7.23), (7.24) and the fact that s_{k_l} is rejected, we obtain

$$(1-\eta_1) < \left|\frac{Ared_k(s_{k_l};\rho_{k_l})}{Pred_k(s_{k_l};\rho_{k_l})} - 1\right| \leq \frac{[K_6 + K_7K_{14} + \alpha K_8K_{14}]\|s_{k_l}\|}{K_{10}}.$$

Hence

$$\|s_{k_l}\| \geq \frac{(1-\eta_1)K_{10}}{K_6 + K_7K_{14} + \alpha K_8K_{14}}. \tag{7.25}$$

Now, using (7.20) and (7.25), we obtain the bound

$$\delta_k \geq \frac{(1-\eta_1)K_{10}\,K_{13}}{K_6 + K_7K_{14} + \alpha K_8K_{14}}.$$

Defining

$$\tilde{\delta} = \min\{\delta_{\min}, K_{12}, \frac{(1-\eta_1)K_{10}\,K_{13}}{K_6 + K_7K_{14} + \alpha K_8K_{14}}\},$$

we obtain the desired bound. □

Now we can show that the nondecreasing sequence of penalty parameters generated by the nonlinear programming Algorithm 6.1 is bounded.

LEMMA 7.12. *Under the standard assumptions, if the algorithm does not terminate then*

$$\lim_{k\to\infty} \rho_k = \rho^\star < \infty. \tag{7.26}$$

Furthermore, there exists a positive integer k_ρ such that $\rho_k = \rho^\star$ for every $k \geq k_\rho$.

Proof. If the penalty parameter at x_k is increased, then from Lemma 7.10 and Lemma 7.11, we have

$$\rho_k \leq \frac{K_{11}}{\delta_k} \leq \frac{K_{11}}{\tilde{\delta}}.$$

Therefore $\{\rho_k\}$ is a bounded sequence, and since it is nondecreasing, there exists $\rho^\star < \infty$ such that

$$\lim_{k\to\infty} \rho_k = \rho^\star.$$

According to the penalty parameter updating scheme, we know that if the penalty parameter is increased, it is increased by a quantity greater than or equal to β, and since the sequence $\{\rho_k\}$ converges to $\rho^\star < \infty$, the number of iterations at which the penalty parameter is increased must be finite. Thus there exists an index k_ρ such that $\rho_k = \rho_{k_\rho} \quad \forall k \geq k_\rho$. This completes the proof. □

This last result will play a crucial role in the proof of the global convergence of Algorithm 6.1.

8. The main global convergence results. This section is devoted to presenting our main global convergence results. We start with the finite termination theorem where we show that the general nonlinear programming algorithm is well-defined. In Section 8.2, we will present more properties of the trust-region radius sequence generated by the algorithm. In Section 8.3, we prove global convergence of our algorithm.

8.1. The finite termination theorem. The following lemma shows that the nonlinear programming Algorithm 6.1 is well-defined in the sense that at each iteration we can find an acceptable step after finite number of trial step computations, or equivalently, trust-region reductions.

THEOREM 8.1. *Under the standard assumptions, unless some iterate x_c satisfies the termination condition of Algorithm 6.1, an acceptable step from x_c will be found after finitely many trust-region subproblem solutions.*

Proof. The proof follows from Theorem 5.1 of El-Alem [8]. □

8.2. Properties of the sequence δ_k. From the way of updating the trust-region radius, $\{\delta_k\}$ is bounded from above. The following lemma shows that the sequence $\{\delta_k\}$ is also bounded from below. *i. e.* it is bounded away from zero.

LEMMA 8.2. *Under the standard assumptions, assume that the algorithm does not terminate. Then there exists a positive constant $\delta_\star$ which does not depend on the iterates such that for all k,*

$$\delta_k \;\geq\; \delta_\star. \tag{8.1}$$

Proof. The proof follows by using the same technique as in Lemma 7.11. In the notation of that proof, we have three cases to consider.

i) $s_{k_1} = s_k$, that is, there is no unacceptable step in between; then accordingly to the rule for updating the trust-region radius we have

$$\delta_k \;\geq\; \max\{\delta_{k-1}, \delta_{\min}\} \;\geq\; \delta_{\min}. \tag{8.2}$$

ii) If $s_{k_1} \neq s_k$ and $\|C_k\| > \alpha\delta_{k_i}$ for all $i = 1, \ldots, j$, the proof is exactly the same as before; see ii) in Lemma 7.11, for details.

iii) Now, if $s_{k_1} \neq s_k$ and $\|C_k\| > \alpha\delta_{k_i}$ does not hold for all $i = 1, \ldots, j$, as in Lemma 7.11, let l be the largest index such that $\|C_k\| \leq \alpha\delta_{k_l}$ holds, and from (7.20), we have

$$\delta_k \;\geq\; K_{13}\|s_{k_l}\|. \tag{8.3}$$

Now, since for the iterate k_l, $\|C_k\| \leq \alpha\delta_{k_l}$, then $Pred_k(s_{k_l}; \rho_{k_l}) \geq K_{10}\delta_{k_l}$. Since $|Ared_k(s_{k_l}; \rho_{k_l}) - Pred_k(s_{k_l}; \rho_{k_l})| \leq K_9\rho_{k_l}\|s_{k_l}\|^2$, and because the step s_{k_l} is an unacceptable step, we have

$$(1 - \eta_1) < \left| \frac{Ared_k(s_{k_l}; \rho_{k_l})}{Pred_k(s_{k_l}; \rho_{k_l})} - 1 \right| \leq \frac{K_9\rho_{k_l}\|s_{k_l}\|^2}{K_{10}\delta_{k_l}} \leq \frac{K_9\rho^\star\|s_{k_l}\|}{K_{10}}.$$

The above inequality implies that

$$\|s_{k_i}\| \geq \frac{(1-\eta_1)K_{10}}{K_9 \rho^\star}.$$

This inequality and (8.3) allow us to write

$$\delta_k \geq \frac{(1-\eta_1)K_{10}K_{13}}{K_9 \rho^\star}. \tag{8.4}$$

From (7.14), (8.2) and (8.4), if we let

$$\delta_\star = \min\left\{\tilde{\delta},\ \frac{(1-\eta_1)K_{10}K_{13}}{K_9\rho^\star}\right\},$$

we obtain $\delta_k \geq \delta_\star$, which is the desired result. □

8.3. The global convergence theorem. Now we present our main global convergence result. Namely, under the standard assumptions, the general nonlinear programming algorithm generates a sequence of iterates $\{x_k\}$, which has at least a subsequence that converges to a stationary point of problem (EQC).

THEOREM 8.3. *Under the standard assumptions, given any $\varepsilon_{tol} > 0$, the algorithm terminates because*

$$\|W_k^T \nabla_x \ell_k\| + \|C_k\| < \varepsilon_{tol}. \tag{8.5}$$

Proof. Let us assume that the algorithm does not terminate. We prove (8.5) by contradiction. We begin by assuming that the sequence $\{\|C_k\|\}$ is bounded away from zero. Then there exists $\tau > 0$ such that for all $k, \|C_k\| \geq \tau$. Let $\bar{k}$ be some iterate such that $\bar{k} \geq k_\rho$, where k_ρ is as in Lemma 7.12. For all $x \in \Omega$, we have

$$\|C(x)\| \geq \|C(x_{\bar{k}})\| - \|C(x) - C(x_{\bar{k}})\| \geq \|C(x_{\bar{k}})\| - \nu_2\|x - x_{\bar{k}}\|.$$

Define $\sigma = \frac{\|C(x_{\bar{k}})\|}{2\nu_2}$ and consider a ball $\mathcal{U}$ of center $x_{\bar{k}}$ and radius σ. Then for all $x \in \mathcal{U}$, we have

$$\|C(x)\| \ \geq\ \frac{1}{2}\|C(x_{\bar{k}})\|. \tag{8.6}$$

Now let us assume that all $x_j \in \mathcal{U},\ j \geq \bar{k}$. Inequalities (5.8), (7.2), and (8.6), and the fact that $\|C_{\bar{k}}\| \geq \tau$ and $\delta_j \geq \delta_\star$ allow us to write, for all $j \geq \bar{k}$

$$\begin{aligned} Pred_j &\geq \frac{\rho_j}{2}[\|C_j\|^2 - \|C_j + \nabla C_j^T s_j\|^2] \geq \frac{K_2}{2}\|C_j\| \min\{K_3\|C_j\|,\ r\delta_j\} \\ &\geq \frac{K_2}{4}\|C_{\bar{k}}\| \min\{\frac{K_3}{2}\|C_{\bar{k}}\|,\ r\delta_j\} \geq \frac{K_2\tau}{4}\min\left\{\frac{K_3\tau}{2},\ r\delta_\star\right\} \\ &\geq K_{15} > 0. \end{aligned}$$

Since an acceptable step can be found in a finite number of iterations and for all $x_j \in \mathcal{U},\ \ j \geq \bar{k}$,

$$\mathcal{L}_j - \mathcal{L}_{j+1} = Ared_j \geq \eta_1 Pred_j \geq\ \eta_1 K_{15}. \tag{8.7}$$

So there are infinitely many iterates for which $\mathcal{L}$ is decreased by a positive quantity. This gives a contradiction to the fact that, under the standard assumptions and because of Lemma 7.12, $\mathcal{L}(x;\rho)$ is bounded below on Ω and means that $\{x_k\}$ must leave the ball.

Consider the iterates outside the ball $\mathcal{U}$. Let $i+1$ be the first iterate that leaves the ball. Since $x_{i+1} \neq x_{\bar{k}}$, there exists at least one acceptable step from iteration $\bar{k}$ through iteration $i+1$. For those iterates, we have

$$\mathcal{L}_{\bar{k}} - \mathcal{L}_{i+1} = \sum_{p=\bar{k}}^{i}(\mathcal{L}_p - \mathcal{L}_{p+1}) = \sum_{p=\bar{k}}^{i} Ared_p \geq \sum_{p=\bar{k}}^{i} \eta_1 Pred_p. \tag{8.8}$$

Now, by using (8.7) we have

$$\mathcal{L}_{\bar{k}} - \mathcal{L}_{i+1} \geq \eta_1 K_{15} = K_{16} > 0. \tag{8.9}$$

Since the sequence $\{\mathcal{L}_k\}$ is bounded and decreasing, it has a limit $\mathcal{L}_\star$. Now if we take the limit as i goes to infinity, we obtain

$$\mathcal{L}_{\bar{k}} - \mathcal{L}_\star \geq K_{16} > 0. \tag{8.10}$$

Now contradiction would arise if we take the limit as $\bar{k}$ goes to infinity. Thus the assumption $\|C_k\| \geq \tau$ can not hold. Therefore there exists a subsequence $\{k_i\}$ such that

$$\lim_{k_i \to \infty} \|C_{k_i}\| \;=\; 0. \tag{8.11}$$

Now let us suppose that $\|W_k^T \nabla_x \ell_k\| \geq \tau_1$. Since $\|C_{k_i}\|$ goes to zero and the sequence of trust-region radii is bounded away from zero, there exists an index $N_1 > 0$ such that for all $k_i \geq N_1$, $\|C_{k_i}\| \leq \alpha\delta_{k_i}$, with α as in (7.8). Then from Lemma 7.8, we have

$$Pred_{k_i} \geq K_{10}\delta_{k_i} \geq K_{10}\delta_\star > 0. \tag{8.12}$$

On the other hand, because of Theorem 8.1, for any iterate k an acceptable step (and hence new iterate $k+1$) is found. In addition to this, if for any $j \in \{k_i\}$ there is no $j+1 \in \{k_i\}$, this will contradict (8.11). So, for any $j \in \{k_i\}$ and $j \geq k_\rho$, where k_ρ is as in Lemma 7.12, we have

$$\mathcal{L}_j - \mathcal{L}_{j+1} \geq Ared_j \geq \eta_1 Pred_j \geq K_{10}\delta_\star > 0. \tag{8.13}$$

Again a contradiction will arise if we take the limit as j goes to infinity. So

$$\lim_{k_i \to \infty} \|W_{k_i}^T \nabla_x \ell_{k_i}\| \;=\; 0. \tag{8.14}$$

Then from (8.11) and (8.14) we have

$$\liminf_{k \to \infty} [\|W_k^T \nabla_x \ell_k\| + \|C_k\|] = 0.$$

This contradicts the assumption that the algorithm does not terminate and implies that the algorithm terminates for any given $\varepsilon_{tol} > 0$. This completes the proof. □

9. An example algorithm. In this section we propose, as an example, a particular step choice algorithm for **step 2** of Algorithm 6.1. We include different ways for computing s_c^n accordingly to the dimension of the problem. We will then state the complete algorithm for finding the trial step. Finally, in Sections 9.5 and 9.6 we will show that the trial step generated by this algorithm satisfies the pair of fraction of Cauchy decrease conditions and (5.1).

The step choice algorithm we propose in this section is based on a conjugate directions method. It can be viewed as a generalization of the Steihaug-Toint dogleg algorithm for the unconstrained problem. This algorithm is much like a trust-region version of an algorithm due to Nash [17].

9.1. The Steihaug-Toint dogleg algorithm. This section is devoted to describing the generalized dogleg algorithm introduced by Steihaug [25] and Toint [28], for approximating the solution of problem (TRS), (see Section 2). This algorithm is based on the linear conjugate gradient method.

ALGORITHM 9.1. **Steihaug-Toint dogleg algorithm for (TRS)**

Given x_c and δ_c.

step 0: *(Initialization)*
Set $\hat{s}_0 = 0$.
Set $r_0 = -(G_c\hat{s}_0 + \nabla f_c)$.
If $r_0 = 0$ *then* **terminate.**
Set $d_0 = r_0$.
Set $i = 0$.

step 1: *Compute* $\gamma_i = d_i^T G_c d_i$.
If $\gamma_i > 0$ *then go to* **step 2:**
Otherwise (* d_i *is a direction of negative or zero curvature* *)
compute $\tau > 0$ *such that* $\|\hat{s}_i + \tau d_i\| = \delta_c$.
Set $s_c = \hat{s}_i + \tau d_i$ *and* **terminate.**

step 2: *Compute* $\alpha_i = \frac{\|r_i\|^2}{\gamma_i}$.
Set $\hat{s}_{i+1} = \hat{s}_i + \alpha_i d_i$.
If $\|\hat{s}_i\| < \delta_c$ *go to* **step 3:**
Otherwise (* *the step is too long, take the dogleg step* *)
compute $\tau > 0$ *such that* $\|\hat{s}_i + \tau d_i\| = \delta_c$.
Set $s_c = \hat{s}_i + \tau d_i$ *and* **terminate.**

step 3: *Compute* $r_{i+1} = r_i - \alpha_i G_c d_i$.

$$\text{If } \frac{\|r_{i+1}\|}{\|\nabla f_c\|} \leq \xi_c, \ (0 < \xi_c \leq \xi < 1) \text{ then}$$

set $s_c = \hat{s}_{i+1}$ *and* **terminate.**

step 4: *Compute* $\beta_i = \frac{\|r_{i+1}\|^2}{\|r_i\|^2}$.
Set $d_{i+1} = r_{i+1} + \beta_i d_i$.
Set $i = i + 1$ *and go to* **step 1:**

The Steihaug-Toint dogleg algorithm is well-known for being suitable for large-scale unconstrained problems. It can be used in the framework of any general trust-region algorithm for solving problem (UCMIN).

9.2. Computing a relaxed normal component. We start our proposed step choice algorithm by finding a relaxed normal component s_c^n of the trial step. This step must satisfy a fraction of Cauchy decrease condition on the constraint norm inside the inner trust region. It determines for us which translate of the null space of the constraint Jacobian will be the one in which we choose the tangent component s_c^t.

We repeat, because it is so important, that we do not require that s_c^n be normal to the tangent space, just that it satisfies (5.1). In fact, below we will see that one way we might choose the relaxed normal component by finding a linearly feasible point and just scaling it back onto the inner trust region.

9.2.1. Via Craig's algorithm. First we note that we can solve for a linearly feasible point by using Craig's algorithm on the underdetermined linear system $\nabla C_c^T s + C_c = 0$. Craig's algorithm consists of making the transformation $s = \nabla C_c y$ and applying the standard conjugate gradient algorithm to the following $m \times m$ linear system

$$\nabla C_c^T \nabla C_c y + C_c = 0.$$

This implies that

$$s_c^{\mathsf{craig}} = s_c^{\mathsf{mn}} = -\nabla C_c(\nabla C_c^T \nabla C_c)^{-1} C_c.$$

Furthermore, the result is the constraint normal and it requires no more than m iterations. Preconditioning is very important of course, but it certainly will depend on the particular application.

Therefore, we can find the step s_c^n by a Steihaug-Toint version of Craig's algorithm in the inner trust region of radius $r\delta_c$. In this algorithm, iterates will be generated until we find the desired constraint normal s_c^{mn} such that $||s_c^{\mathsf{mn}}|| \leq r\delta_c$ or until s_j^{craig} and s_{j+1}^{craig} straddle the $r\delta_c$ trust-region boundary. For the first case, we set $s_c^n = s_c^{\mathsf{mn}}$. For the second case, we choose the dogleg step: $s_c^{\mathsf{dog}} \in [s_j^{\mathsf{craig}}, s_{j+1}^{\mathsf{craig}}] \cap \{s : ||s|| = r\delta_c\}$ and set $s_c^n = s_c^{\mathsf{dog}}$.

It is clear that $s_c^n = s_c^{\mathsf{dog}}$ satisfies the fraction of Cauchy decrease condition required by **step 2** of Algorithm 6.1. It is not difficult to prove that each Craig iterate is the normal to the subspace of the tangent space spanned by the steps up to that point and that each $\{s_j^{\mathsf{craig}}\}$ satisfies (5.1). Hence, by convexity, s_c^{dog} satisfies (5.1).

9.2.2. Via a linearly feasible point. There are some problems for which Craig's method might be too slow and too hard to precondition to use the "inner Steihaug-Toint" algorithm given above. Or, for reasons too technical to be of much interest here, someone might prefer to do an implementation that computes a linearly feasible point s_c^{lf} either by Craig's method or by some special application dependent methods. The point of this subsection is that when this is the case, s_c^n can be taken to be the projection of s_c^{lf} back onto the inner trust region. If s_c^{lf} satisfies (5.1), then so does s_c^n.

Suppose we have any linearly feasible point s_c^{lf} that satisfies (5.1). Then, if it is inside the inner trust region, we can take s_c^n to be that point and it clearly satisfies the fraction of Cauchy decrease condition required by **step 2** of Algorithm 6.1. If $||s_c^{\mathsf{lf}}|| \geq r\delta_c$, then we take

$$s_c^n = \frac{r\delta_c}{||s_c^{\mathsf{lf}}||} \cdot s_c^{\mathsf{lf}} \ .$$

A classical mathematical programming way to compute a linearly feasible point that encompasses some special purpose methods we have seen for some inverse problems is as follows. In some way, divide s into so-called basic and nonbasic components.

Let us assume that we have done so, and using column pivoting, we write ∇C^T as $\nabla C^T = [B|N]$ where B is a nonsingular matrix corresponding to the basic components of s. This corresponds to $W_c = \begin{bmatrix} -B_c^{-1}N_c \\ I_{n-m} \end{bmatrix}$. Now since

$$\nabla C_c^T s = B_c s_B + N_c s_N = -C_c,$$

we have

$$s_B = -B_c^{-1}(C_c + N_c s_N),$$

and then if we choose $s_N = 0$ and $s_B = -B_c^{-1}C_c$, a feasible point will be

$$s_c^{\mathrm{lf}} = (s_B, s_N)^T = (-B_c^{-1}C_c, 0)^T.$$

As long as $\{\|B_k^{-1}\|\}$ is uniformly bounded by some constant γ_* (see Lemma 9.3 for a proof), s_c^{lf} satisfies (5.1) where the constant here is γ_*.

9.3. Computing the tangential component. We now assume that we have the relaxed normal component step s_c^n. We start the process of computing the tangent space component s_c^t by formatting the basis matrix $W_c \in \Re^{n\times(n-m)}$. The columns of W_c form a basis to the null space of the constraints $\mathcal{N}(\nabla C_c^T)$.

We then transfer the constrained problem into an unconstrained trust-region problem of dimension $n-m$, in the following form:

$$\begin{cases} \textit{minimize} & \frac{1}{2}\bar{s}^{tT}\bar{H}_c\bar{s}^t + \nabla q_c(s_c^n)^T W_c \bar{s}^t + q(s_c^n) \\ \textit{subject to} & \|W_c\bar{s}^t + s_c^n\| \le \delta_c, \end{cases}$$

where $\bar{s}_c^t \in R^{n-m}$, and set $s_c^t = W_c\bar{s}_c^t$. The step s_c^t is the component in the tangent space of the constraints and the matrix $\bar{H}_c = W_c^T H_c W_c \in \Re^{(n-m)\times(n-m)}$ is the reduced Hessian matrix. Now we use the Steihaug-Toint algorithm to determine $\bar{s}_c^t$ such that $\|W_c\bar{s}^t + s_c^n\| \le \delta_c$.

The complete algorithm for finding the trial step is presented in the following section

9.4. Conjugate reduced gradient algorithm for EQC. Here we write, in more detail, the example algorithm for computing a trial step.

ALGORITHM 9.2. *Given $x_c \in \Re^n$ and $\delta_c > 0$,*

I. *FEASIBILITY:*

1) *If x_c is feasible go to* **II.**

2) *Determine s_c^n. (* Use, for example, $s_c^n = s_c^{\mathrm{dog}}$ or $s_c^n = \frac{r\delta}{\|s_c^{\mathrm{lf}}\|}s_c^{\mathrm{lf}}$ and $s_c^{\mathrm{lf}} = (-B_c^{-1}C_c, 0)^T$. *)*

II. *MINIMIZATION:*

(Find s_c by applying the* **<u>CRG/Steihaug-Toint algorithm</u>**, *to*

$$\begin{cases} \textit{minimize} & q_c(s) \\ \textit{subject to} & \nabla C_c^T(s - s_c^n) = 0 \\ & \|s\| \le \delta_c. \end{cases}$$

*starting from $s = s_c^n$ *)*

step 0: *(Initialization)*
Set $\hat{s}_0 = s_c^n$.
Set $r_0 = -W_c^T(H_c s_c^n + \nabla \ell_c)$.
If $r_0 = 0$ *then* **terminate.**
Set $d_0 = r_0$.
Set $i = 0$.
step 1: *Compute* $\gamma_i = d_i^T H_c d_i$.
If $\gamma_i > 0$ *then go to* **step 2:**,
otherwise (* d_i *is a direction of negative or zero curvature* *)
compute $\tau > 0$ *such that* $\|\hat{s}_i + \tau d_i\| = \delta_c$.
Set $s_c = \hat{s}_i + \tau d_i$ *and* **terminate.**
step 2: *Compute* $\alpha_i = \frac{\|r_i\|^2}{\gamma_i}$.
Set $\bar{s}_{i+1} = \bar{s}_i + \alpha_i d_i$.
If $\|\bar{s}_i\| < \delta_c$ *go to* **step 3:**,
otherwise (* *the step is too long, take the dogleg step* *)
compute $\tau > 0$ *such that* $\|\hat{s}_i + \tau d_i\| = \delta_c$.
Set $s_c = \hat{s}_i + \tau d_i$ *and* **terminate.**
step 3: *Compute* $r_{i+1} = r_i - \alpha_i H_c d_i$.

$$\text{If } \frac{\|r_{i+1}\|}{\|\nabla \ell_c\|} \le \xi_c, \ (0 < \xi_c \le \xi < 1) \text{ then}$$

set $s_c = \hat{s}_{i+1}$ *and* **terminate.**
step 4: *Compute* $\beta_i = \frac{\|r_{i+1}\|^2}{\|r_i\|^2}$.
Set $d_{i+1} = r_{i+1} + \beta_i d_i$.
Set $i = i + 1$ *and go to* **step 1:**

9.5. Sufficient decrease by the steps. In this section we show that the conjugate reduced gradient algorithm produces steps that satisfy the conditions we impose on the steps in **step 2** of Algorithm 6.1. In particular, we show that both the relaxed normal and the tangential components of the trial steps satisfy their respective fraction of Cauchy decrease conditions.

The following Lemma gives a bound on the matrix $B(x)^{-1}$.

LEMMA 9.3. *Under the standard assumptions, the matrix* $B(x)^{-1}$ *is bounded for all* $x \in \Omega$.

Proof. The proof follows from the continuity of $B(x)$ for all $x \in \Omega$ and using the well-known Banach Perturbation Lemma (see, for example, Ortega and Rheinboldt [19], page (46)). □

The following Lemma gives a bound on the reducer matrix W_k.

LEMMA 9.4. *Under the standard assumptions, the reducer matrix*

$$W(x) = \begin{bmatrix} -B(x)^{-1}N(x) \\ I_{n-m} \end{bmatrix}$$

is bounded for all $x \in \Omega$.

Proof. For all $x \in \Omega$, $\|\nabla C(x)^T\| = \| [B(x) \mid N(x)] \| \le \nu_2$. Clearly, $B(x)$ and $N(x)$ are bounded, that is, there exist constants $\nu_B, \nu_N > 0$ such that the Frobenius norms of $B(x)$ and $N(x)$ are bounded. i. e., $\|B(x)\|_F \le \nu_B$ and $\|N(x)\|_F \le \nu_N$. Now, consider:

$$\begin{aligned} \|W(x)\|_F^2 &= tr(W(x)^T W(x)) = tr[(B(x)^{-1}N(x))^T B(x)^{-1}N(x)] + tr(I_{n-m}) \\ &= \|B(x)^{-1}N(x)\|_F^2 + n - m. \end{aligned}$$

Lemma 9.3 allows us to write $||B(x)^{-1}|| \leq \nu_{B^{-1}}$. Hence $||W(x)|| \leq \sqrt{\nu_{B^{-1}}^2 \nu_N^2 + n - m}$ and we obtain the desired result. $\square$

The following lemma shows that the relaxed normal component s_c^n, satisfies a fraction of Cauchy decrease condition on the quadratic model of the linearized constraints.

LEMMA 9.5. *Let s_k be a step generated by Algorithm 9.2 at the current iterate. Then s_k satisfies a fraction of Cauchy decrease condition on the quadratic model of the linearized constraints, i.e,*

$$||C_k||^2 - ||C_k + \nabla C_k^T s_k||^2 \geq K_2 ||C_k|| \min\{r\delta_k \ , \ K_3||C_k||\}, \tag{9.1}$$

where K_2 and K_3 are constants independent of the iterates.

Proof. We first prove that

$$||\nabla C_c^T s_c + C_c|| \leq ||\nabla C_c^T s_c^{\mathsf{cp}} + C_c||. \tag{9.2}$$

If x_c is feasible, (9.1) is valid a fortiori.
Assume $C_c \neq 0$. We consider three cases:

Case 1: If $s_c^n = s_c^{\mathsf{mn}}$ with $||s^{\mathsf{mn}}|| \leq r\delta_c$, then, because s_c^{mn} solves a minimum norm subproblem, we have $||\nabla C_c^T s_c^{\mathsf{mn}} + C_c|| \leq ||\nabla C_c^T s_c^{\mathsf{cp}} + C_c||$.

Case 2: Suppose that we are applying the conjugate reduced gradient algorithm to find s_c^n. Let $\{s_1, s_2, \ldots\}$ be the sequence of iterates generated by the algorithm, hence for all i.

$$s_i = arg\ min\{||\nabla C_c^T s + C_c||, \ s \in span\{p_1, \ldots, p_i\}\}.$$

Assume that $||s_i|| \leq r\delta_c$ and $||s_{i+1}|| \geq r\delta_c$. Therefore

$$s_c^{\mathsf{dog}} = \alpha_1 s_i + (1 - \alpha_1) s_{i+1} \quad \text{with} \quad \alpha_1 \in [0, 1].$$

It is easy to see that

$$||\nabla C_c^T s_i + C_c|| \leq ||\nabla C_c^T s_c^{\mathsf{cp}} + C_c||,$$

$$||\nabla C_c^T s_{i+1} + C_c|| \leq ||\nabla C_c^T s_c^{\mathsf{cp}} + C_c||.$$

By convexity,

$$||\nabla C_c^T s_c^{\mathsf{dog}} + C_c|| \leq ||\nabla C_c^T s_c^{\mathsf{cp}} + C_c||.$$

Case 3: Now suppose that s_c^n is given by $s_c^n = \gamma s_c^{\mathsf{mn}}$ with $\gamma = \frac{r\delta_c}{||s_c^{\mathsf{mn}}||}$ and $\gamma \leq 1$.

The boundedness of γ from below follows directly from the above lemma. We have

$$||\nabla C_c^T s_c^n + C_c|| \leq ||\nabla C_c^T s_c^{\mathsf{cp}} + C_c||.$$

For a proof see Lemma 5.2 of El-Alem [9].

For the above three cases, we note that since

$$\begin{aligned} ||\nabla C_c^T s_c^n + C_c|| &\leq ||C_c|| = ||\nabla C_c^T \cdot 0 \cdot s_c^{\mathsf{mn}} + C_c|| \\ \text{and } ||\nabla C_c^T s_c^n + C_c|| &\geq ||\nabla C_c^T \cdot 1 \cdot s_c^{\mathsf{mn}} + C_c|| \ . \end{aligned}$$

Therefore, we have for some $\gamma_1 \in [0,1]$,

$$\begin{aligned}
\|\nabla C_c^T s_c^n + C_c\| &= \|\nabla C_c^T \gamma_1 s_c^{\mathrm{mn}} + C_c\| \\
&= \|\nabla C_c^T \gamma_1 s_c^{\mathrm{mn}} + \gamma_1 C_c + C_c - \gamma_1 C_c\| \\
&= \|\gamma_1(\nabla C_c^T s_c^{\mathrm{mn}} + C_c) + (1-\gamma_1) C_c\| \\
&\le \gamma_1 \|\nabla C_c^T s_c^{\mathrm{mn}} + C_c\| + (1-\gamma_1)\|C_c\| \\
&\le \gamma_1 \|\nabla C_c^T s_c^{\mathrm{cp}} + C_c\| + (1-\gamma_1)\|C_c\|.
\end{aligned}$$

The rest of the proof then follows from Lemma 6.1 of El-Alem [8]. □

The following lemma shows that the null-space component s_c^t, satisfies a fraction of Cauchy decrease condition on the quadratic model of the Lagrangian.

LEMMA 9.6. *Let s_k be a trial step generated by the algorithm, then*

$$q_k(s_k^n) - q_k(s_k) \ \ \ge \ \ \frac{1}{2}\|W_k^T \nabla q_k(s_k^n)\| \min\left\{ \frac{\|W_k^T \nabla q_k(s_k^n)\|}{\|W_k^T H_k W_k\|},\ (1-r)\delta_k \right\}.$$

Proof. Since we are solving the reduced problem

$$\begin{cases} \textit{minimize} & \frac{1}{2}\bar{s}^{tT} \bar{H} \bar{s}^t + \nabla q_c(s_c^n)^T W_c \bar{s}^t + q(s_c^n) \\ \textit{subject to} & \|\bar{s}^t + s_c^n\| \le \delta_c, \end{cases}$$

which is an unconstrained trust-region subproblem. Then using Theorem 2.5 of Steihaug [25], the proof follows immediately. □

LEMMA 9.7. *Let s_c be a trial step generated by the algorithm. Then there exists a positive constant K_4, which does not depend on x_c such that*

$$q_c(s_c^n) - q_c(s_c) \ \ \ge \ \ \frac{1}{2}\|W_c^T \nabla q_c(s_c^n)\| \min\left\{ K_4 \|W_c^T \nabla q_c(s_c^n)\|,\ (1-r)\delta_c \right\}.$$

Proof. The proof follows from the above lemma and Lemma 9.4. □

9.6. The relaxed normal component and (5.1). In the following lemma, we show that the relaxed normal component obtained by our proposed step choice algorithm satisfies inequality (5.1).

LEMMA 9.8. *The relaxed normal component computed by our proposed step choice algorithm satisfies*

$$\|s_k^n\| \le K_1 \|C_k\|,$$

where K_1 is a positive constant independent of k.

Proof. The proof is given with the discussion of how to compute a relaxed normal component. See Section 9.2. □

10. Discussion and concluding remarks. We have established a global convergence theory for a broad class of nonlinear programming algorithms for the smooth problem with equality constraints. The class includes algorithms based on the full-space approach and the tangent-space approach. The family is characterized by generating steps that satisfy very mild conditions on the normal and tangential components. The normal component satisfies a fraction of Cauchy decrease condition on the quadratic model of the linearized constraints and the tangential component satisfies a fraction of Cauchy decrease condition on the quadratic model of the Lagrangian function associated with the problem, reduced to the tangent space of the constraints. Of course the step, which is the sum of these components, satisfies both conditions.

The augmented Lagrangian was chosen as a merit function. The scheme for updating the penalty parameter is the one proposed by El-Alem [8] since it ensures that the merit function is decreased at each iteration by at least a fraction of Cauchy decrease. This indicates compatibility with the fraction of Cauchy decrease conditions imposed on the trial steps.

The algorithm was proved to be well-defined, in the sense that at each iterate an acceptable step can be found after solving a finite number of trust region subproblems. Because of the properties of the step and the way that the trust-region radii are updated, we were able to prove that the sequence of trust region radii is bounded away from zero. This result together with the way that the penalty parameter is chosen allowed us to prove that the sequence of penalty parameter is increased only finitely many times. The global convergence analysis was constructed on these results.

In presenting the algorithm, we have left open the way of computing the trial steps to satisfy the double fraction of Cauchy decrease condition. This will allow the inclusion of a wide variety of trial step calculation techniques. For the same reason we have left the way of updating the Lagrange multiplier vector and the Hessian matrix undefined.

With respect to the trial steps, we have suggested an algorithm of the class that should work quite well for large problems. The algorithm is a generalization of the Steihaug-Toint dogleg algorithm for the unconstrained case.

The projection formula for the multiplier can be used as a scheme for updating the multiplier since it fits the condition imposed on the multiplier updating scheme. Namely, under the standard assumptions, it produces bounded multipliers. For large problems, $\lambda = -B^{-1}\nabla f$ can be used as an update formula. This will match better with the reducer matrix W, specially for problems where B can be easily identified. See Dennis and Lewis [5]. In either case, the uniform boundedness of $\{\lambda_k\}$ follows from our global assumptions.

The exact Hessian matrix can be perhaps gotten by using automatic differentiation. See Bischof *et al.* [1]. However, an approximation to the Hessian of the Lagrangian can be used. Also, for example, setting H_k to a fixed matrix (*e. g.* $H_k = 0$) for all k is valid. The question of how to use a secant approximation of the Hessian of the Lagrangian in order to produce a more efficient algorithm is a research topic. We believe that Tapia [27] will be of considerable value here.

A related question that has to be looked at is the search for preconditioners to produce more efficient algorithms. We believe that the reducer matrix W should play a role in that search. See Dennis and Lewis [5].

For future work, there are some questions that we would like to answer:

Currently, the local analysis of this class of algorithms is being studied and a preliminary implementation of the algorithm based on the CRG/Steihaug-Toint via a tangent-space approach has to be completed. An efficient implementation should be based on the right selection of the submatrix B in the CRG-algorithm.

This theory is developed for the equality constrained case, but it can be applied to the general case, by one of the strategies known as EQP and IQP. Here, we mean that in the EQP strategy the choice of the active set is made outside the algorithm that determines the step while in the IQP strategy, that choice is made inside the procedure that determines the step. Since the active set may change at each iteration, the choice of the submatrix B, will be strongly affected. Certainly, this is an important topic that deserves to be investigated.

11. Acknowledgements. We wish to thank Richard Byrd for many helpful comments and the referees for pointing out many unclear points in the original submitted manuscript.

REFERENCES

[1] C. BISCHOF, A. CARLE, G. CORLISS, A. GRIEWANK, and P. HOVLAND. Adifor – generating derivative codes from fortran programs. *Scientific Programming*, 1(1):11–29, 1992.

[2] R. H. BYRD, R. B. SCHNABEL, and G. A. SHULTZ. A trust region algorithm for nonlinearly constrained optimization. *SIAM J. Numer. Anal.*, 24:1152–1170, 1987.

[3] R. CARTER. *Multi-model algorithms for optimization*. PhD thesis, Department of Mathematical Sciences, Rice University, Houston, Texas, 1986.

[4] M. R. CELIS, J. E. DENNIS Jr., and R. A. TAPIA. A trust region strategy for nonlinear equality constrained optimization. In *Numerical Optimization 1984*. SIAM, Philadelphia, Pennsylvania, 1985.

[5] J. E. DENNIS Jr. and R. M. LEWIS. On the incorporation of decomposition methods for pde in the formulation of nonlinear programming problems. Technical report, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, 1992. in preparation.

[6] J. E. DENNIS Jr. and R. B. SCHNABEL. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, New Jersey, 1983. Russian edition, Mir Publishing Office, Moscow, 1988, O. Burdakov, translator.

[7] M. M. EL-ALEM. *A global convergence theory for a class of trust region algorithms for constrained optimization*. PhD thesis, Department of Mathematical Sciences, Rice University, Houston, Texas, 1988.

[8] M. M. EL-ALEM. A global convergence theory for the Celis-Dennis-Tapia trust region algorithm for constrained optimization. *SIAM J. Numer. Anal.*, 28:266–290, 1991.

[9] M. M. EL-ALEM. A robust trust region algorithm with non monotonic penalty parameter scheme for constrained optimization. Technical report, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, 1992. Submitted for publication.

[10] R. FLETCHER. An exact penalty function for nonlinear programming with inequalities. *Mathematical Programming*, 5:129–150, 1973.

[11] R. FLETCHER. *Practical Methods of Optimization*. John Wiley & Sons, New York, 1987.

[12] P. GILL, W. MURRAY, M. SAUNDERS, and M. WRIGHT. Some theoretical properties of an augmented Lagrangian merit function. Technical Report Rept. SOL 86-6, Stanford University, Stanford, CA, 1986.

[13] P. GILL, W. MURRAY, and M. WRIGHT. *Practical Optimization*. Academic Press, Inc, New York, 1981.

[14] M. C. MACIEL. *A global convergence theory for a general class of trust region algorithm for equality constrained optimization*. PhD thesis, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, 1992.

[15] J. J. MORÉ. Recent developments in algorithms and software for trust region methods. In A. Bachem, M. Grotschel, and B. Korte, editors, *Mathematical programming. The state of the art*, pages 258–287. Springer-Verlag, New York, 1983.

[16] J. J. MORÉ and D. C. SORENSEN. Computing a trust region step. *SIAM J. Sci. Stat. Comput.*, 4:553–572, 1983.

[17] S. G. NASH. Preconditioning of truncated-Newton methods. *SIAM J. Sci. Stat. Computing*, 6(3):599–616, 1985.

[18] E. O. OMOJOKUN. *Trust region strategies for optimization with nonlinear equality and inequality constraints*. PhD thesis, Department of Computer Science, University of Colorado, Boulder, Colorado, 1989.

[19] J. M. ORTEGA and W. C. RHEINBOLDT. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York, 1970.

[20] M. J. D. POWELL. Convergence properties of a class of minimization algorithms. In O.L. Mangasarian, R.R. Meyer, and S.M. Robinson, editors, *Nonlinear Programming 2*, pages 1–27. Academic Press, New York, 1975.

[21] M. J. D. POWELL and Y. YUAN. A trust region algorithm for equality constrained optimization. *Math. Prog.*, 49:189–211, 1991.

[22] K. SCHITTKOWSKI. On the convergence of a sequential quadratic programming method with an augmented Lagrangian line search functions. *Math. Operationsforchung Und Statistik Ser. Optimization*, 14:197–216, 1983.

[23] G. A. SHULTZ, R. B. SCHNABEL, and R. H. BYRD. A family of trust-region-based algorithms for unconstrained minimization with strong global convergence properties. *SIAM J. Numer. Anal.*, 22:47–67, 1985.

[24] D. SORENSEN. Newton's method with a model trust region modification. *SIAM J. Num. Anal.*, 19:409–426, 1982.

[25] T. STEIHAUG. The conjugate gradient method and trust regions in large scale optimization. *SIAM J. Num. Anal.*, 20:626–637, 1983.

[26] R. A. TAPIA. Quasi-Newton methods for equality constrained optimization: equivalence of existing methods and a new implementation. In O.L. Mangasarian, R.R. Meyer, and S.M. Robinson, editors, *Nonlinear Programming 3*, pages 125–164. Academic Press, New York, 1978.

[27] R. A. TAPIA. On secant update for use in general constrained optimization. *Mathematics of computations.*, 51:181–202, 1988.

[28] PH. L. TOINT. Towards an efficient sparsity exploiting Newton method for minimization. Technical Report 80/4, Departément de Mathématique, Facultés Universitaires de Namur, Belgium, 1980.

[29] A. VARDI. *Trust region strategies for unconstrained and constrained minimization.* PhD thesis, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, New York, 1980.

[30] A. VARDI. A trust region algorithm for equality constrained minimization: convergence properties and implementation. *SIAM J. Numer. Anal.*, 22:575–591, 1985.

[31] K. A. WILLIAMSON. *A robust trust region algorithm for nonlinear programming.* PhD thesis, Department of Mathematical Sciences, Rice University, Houston, Texas 77251-1892, 1990.