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

John E. Dennis
Mahmoud El-Alem
Maria Cristina Maciel

September 1992
(revised August 1995)

TR92-28

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 quasi-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 quasi-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 quasi-normal direction to the tangent space of the constraints and then does feasible conjugate reduced-gradient steps to solve the reduced 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 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, and the REDI Foundation.

‡ Department of Computational and Applied Mathematics & Center for Research on Parallel Computation, 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. It is important to note that the condition is not required on the truely normal component of the trial step, instead it is on the quasi-normal component s_c^n, which is allowed to satisfy the relaxed condition that $\|s_c^n\|_2 \leq K_1\|C(x_c)\|_2$ for some independent constant K_1. The conditions are that the quasi-normal component satisfies a *fraction of Cauchy decrease* condition on the quadratic model of the linearized constraints, and that the tangential component (as measured from the quasi-normal 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 [9].

Conditions 1, and 3 are satisfied by the algorithms of Byrd, Schnabel, and Shultz [2], Celis, Dennis, and Tapia [4], Byrd and Omojokun [21], and Powell and Yuan [23]. Byrd, Schnabel, and Shultz and Byrd and Omojokun require a normal, rather than just a quasi-normal s_c^n, in 2.

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, and we will use the same symbol 0 to denote the real number zero and the zero vector.

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 was one we had in mind as motivation for the convergence theory. It 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 $f : \Re^n \to \Re$ 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:*

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

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 [18] and the book of Dennis and Schnabel [7].

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

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

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

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

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 [18], Powell [22], and Shultz, Schnabel and Byrd [25].

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 [22]. □

We end this section by stating Powell's powerful theorem for unconstrained trust-region algorithms. The proof can be found in Powell [22]. More details about the convergence theory for trust-region algorithms for unconstrained optimization can be found in Fletcher [14], Moré [18], Moré and Sorensen [19], and Sorensen [26].

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:*

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

Notice that this theorem does not prove convergence to a solution to the unconstrained problem, rather it proves a "weak" first order convergence. However, we do not see that as the point of this theorem, nor is it surprising given the weak assumptions on the sequence of local models. In other words, this theorem is not about convergence conditions on a quasi-Newton method. Such a theorem would be expected to be based on analyzing some way of estimating the Hessian, and we all know how important the method for estimating the Hessian is in the practical performance of a trust-region algorithm. In the unconstrained case, the version of Powell's

theorem that says that the sequence of gradients converges to zero, requires the additional hypothesis that the gradient is uniformly continuous. The algorithms here would probably require a uniformly continuous reduced gradient, a strengthening of the assumptions used here. The related algorithms mentioned earlier also prove weak first order stationary convergence, as do we.

The point of this line of research is an analysis of the local quadratic-model/trust-region paradigm for unconstrained optimization. In that context, this theorem says that the power of using a trust-region globalization is that if the first order information is correct, then little is required of the second order information. Specifically, the sequence of model Hessians need only be bounded.

Our theory is analogous for problem (EQC). In this case, the local model of the problem is generally taken to be a linear model of the constraints and a quadratic model of the Lagrangian. The information in the local model depends on the Lagrange multiplier estimates as well as second order information. In this paper, we identify a way to extend the unconstrained paradigm to problem (EQC) for which the only requirement is boundedness of the sequence of model Lagrange multipliers and Hessians.

The above discussion summarizes the point of this paper, which is not to give a convergence proof for a specific SQP approach using a specific Lagrange multiplier estimation technique and perhaps an exact merit function.

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

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

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 [28] 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:

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

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 $\{s : \nabla C_c^T s + C_c = 0\} \cap \{s : \|s\| \leq \delta_c\} = \emptyset$. 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 [17]. See also Byrd, Schnabel and Shultz [2], Celis, Dennis and Tapia [4], Omojokun [21], Powell and Yuan [23], and Vardi [31] and [32].

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

$$\begin{cases} \textit{minimize} & q_c(s^t + s_c^n) \\ \textit{subject to} & \nabla C_c^T s^t = 0 \\ & \|s^t\| \leq \bar{\delta}_c, \end{cases}$$

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 [31],[32] 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, $s_c^n = -\alpha \nabla C_c (\nabla C_c^T \nabla C_c)^{-1} C_c$. 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 $\{s : \nabla C_c^T s + \alpha_\sigma C_c = 0\} \cap \{s : \|s\| \leq \sigma \delta_c\} \neq \emptyset$.

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

Omojokun [21], 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 [18], Moré and Sorensen [19], or Dennis and Schnabel [7].

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 Moore-Penrose 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 [33]. Celis, Dennis, and Tapia [4] choose θ_c based on a *fraction of Cauchy decrease* condition on $\|\nabla C_c^T s + C_c\|^2$. They ask the step to satisfy, for some $r_1 \in (0, 1]$,

$$\|C_c\|^2 - \|C_c + \nabla C_c^T s\|^2 \geq r_1\{\|C_c\|^2 - \|\nabla C_c^T s_c^{\mathsf{cp}} + C_c\|^2\}.$$

This can be done by choosing

$$\theta_c^2 = (\theta_c^{\mathsf{fcd}})^2 \equiv r_1\|\nabla C_c^T s_c^{\mathsf{cp}} + C_c\|^2 + (1 - r_1)\|C_c\|^2 \tag{4.1}$$

where 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_1 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 with associated multiplier estimate λ_c, a step s_c is computed by solving some trust-region subproblems, and a Lagrange multiplier estimate λ_+ is obtained by using some scheme. The point x_+, where $x_+ = x_c + s_c$, is tested using some merit function to decide whether it is a better approximation to a solution $x_\star$. Such merit functions often involve a penalty 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 vectors and the Hessian matrices 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 mild 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 quasi-normal component. We do not require that s_c^n be normal to the tangent space.

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 quasi-normal component gives at least as much decrease as $s_c^{\mathsf{cp}} = -\mathsf{n}_c^{\mathsf{cp}} \nabla C_c C_c$ 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 $\|s\| \leq \hat{\delta}_c$. Equivalently, s_c^n lies in the set

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

where $(\theta_c^{\mathsf{fcd}})^2$ is given by (4.1). Because the quasi-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

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

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) as long as K_1 is greater than a uniform bound on the norm of the right inverse for $\nabla C(x)^T$. When s_c^n is not normal to the tangent space, we do not suggest choosing K_1 and enforcing (5.1). Rather, we suggest (as in Section 9) that (5.1) is enforced naturally by any reasonable algorithm for computing a linearly feasible point.

We will deal with the quasi-normal components of the trial steps assuming that they satisfy (5.1). We are indebted to Robert Michael Lewis for informing us of the effectiveness of this feature in the algorithm which he has implemented to solve a PDE inverse problem [6]. Specifically, this allows special linear algebra developed for simulation constraints to be used in place of prohibitively large least-squares solutions.

Now we use the quasi-normal component to pick a linear manifold $\mathcal{M}_c$ parallel to the null-space of the constraints in which we will select the tangential component. Let $\mathcal{M}_c = \{s : \nabla C_c^T s = \nabla C_c^T s_c^n\}$. Thus, $\mathcal{M}_c \cap \{s = s^t + s_c^n : \|s\| \leq \delta_c\} \neq \emptyset$.

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

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

for some $\sigma > 0$, and

$$(5.2)\quad t_c^{cp} = \begin{cases} \dfrac{\|W_c^T \nabla q_c(s_c^n)\|^2}{\nabla q_c(s_c^n)^T W_c \bar{H}_c W_c^T \nabla q_c(s_c^n)} & \text{if } \dfrac{\|W_c^T \nabla q_c(s_c^n)\|^2 \|W_c W_c^T \nabla q_c(s_c^n)\|}{\nabla q_c(s_c^n)^T W_c \bar{H}_c W_c^T \nabla q_c(s_c^n)} \le \bar{\delta}_c \\ & \text{and } \nabla q_c(s_c^n)^T W_c \bar{H}_c W_c^T \nabla q_c(s_c^n) > 0 \\ \dfrac{\bar{\delta}_c}{\|W_c W_c^T \nabla q_c(s_c^n)\|} & \text{otherwise,} \end{cases}$$

where $\bar{H}_c = W_c^T H_c W_c$ is the reduced Hessian matrix and $\bar{\delta}_c$ is the maximum length of the step allowed inside the set $\mathcal{M}_c \cap \{s = s^t + s_c^n : \|s\| \le \delta_c\}$ in the negative reduced gradient direction $-W_c^T \nabla q_c(s_c^n)$.

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

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

We have intentionally not stated the computation of the tangential component as a trust-region subproblem. Condition 5.2 is a lopsided condition in the sense that $\bar{\delta}_c$ is direction dependent because the quasi-normal step is not the center of the natural trust region for the reduced quadratic. A better step might come from minimizing the reduced quadratic in $\mathcal{M}_c \cap \{s = s^t + s_c^n : \|s\| \le \bar{\delta}_c\}$, and an ideal step would probably come from minimizing the reduced quadratic in $\mathcal{M}_c \cap \{s = s^t + s_c^n : \|s\| \le \delta_c\}$. In any case, both result in steps that satisfy our conditions.

We have defined the tangent space Cauchy step along $-W_c^T \nabla q_c(s_c^n)$, which is the steepest descent direction for $q_c(s_c^n + W_c \bar{s}^t)$ in the ℓ_2 norm. The steepest descent direction in the $\|W_c \cdot\|$ norm would be $-[W_c W_c^T]^{-1} W_c^T \nabla q_c(s_c^n)$. Of course, as long as $[W_c W_c^T]^{-1}$ is uniformly bounded, which seems a reasonable assumption, then either step satisfies a fraction of Cauchy decrease condition with respect to the other, and our theory holds for either. We do not need this boundedness assumption for our choice of Cauchy step. For a particular application, the choice of variables may be determined by which form of the reduced problem is easiest to precondition. See the discussion after Algorithm 9.2. For the problems of interest to us, $-[W_c W_c^T]^{-1} W_c^T \nabla q_c(s_c^n)$ would be an extremely expensive - or impossible - direction to compute.

5.2. Updating the model Lagrange multiplier and the model Hessian. The method for estimating the multiplier λ_c is left unspecified. We only require that the sequence of estimates $\{\lambda_k\}$ be bounded. Any approximation to the Lagrange multiplier vector that produces a bounded sequence can be used. For example, setting λ_k to a fixed vector (or even the zero vector) for all k is valid. Similarly we require only boundedness 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 and Hessian estimates that produce an efficient algorithm. We are addressing some weak assumptions on those estimates $\{\lambda_k\}$ 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 [9], [10] and Powell and Yuan [23].

El-Alem [10] and Powell and Yuan [23] used the formula $\lambda(x) = -(\nabla C(x)^T \nabla C(x))^{-1} \nabla C(x)^T \nabla f(x)$ 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. This means that if ρ is sufficiently large, then the solution to problem (EQC) will be an unconstrained minimizer of the penalty function. See Fletcher [12], [13].

Celis, Dennis, and Tapia [4] and El-Alem [9], on the other hand, with a particular choice of the multiplier, have treated the multiplier as an independent parameter that really only enters in the merit function for accepting the step and updating the other parameters in the algorithm. In other words, one never explicitly uses the merit function in computing the optimization step; it is used only for evaluating the steps. The effect on the trial step computation of the multiplier estimates is in the tangential component through the estimate of the Hessian of the Lagrangian. This is a major difference between merit function roles in trust region algorithms and in line-search algorithms.

In the context of a line-search globalization strategy, Gill, Murray, Saunders, and Wright [15] and Schittkowski [24] have considered the augmented Lagrangian as a merit function, but also as an objective function for choosing the step along the direction of search. They have treated the multiplier as an independent variable and proved global convergence for their algorithms.

In summary, we believe that having an exact penalty function as a merit function is, of course, a desirable property, especially in line-search algorithms. On the other hand, in practice, one never really knows anyway that the penalty constant has been chosen so that the exactness property holds. In [8], [9] global convergence for a particular trust-region method is shown with no assumption of exactness.

In this work, the choice of the multiplier estimate is left open and $\lambda = 0$ is allowed, in which case one is using the ℓ_2 penalty function as a merit function.

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

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

and the *predicted reduction* is defined to be

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

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 [7] or Fletcher [14]. However for our global convergence theory, we use a modification due to Zhang, Kim, and Lasdon [34] (see also El Hallabi and Tapia [11]) of the strategy of updating the trust-region radius. The reader will see that this modification is of no importance in practice; it is merely an analytic formality. At the beginning we set constants $\delta_{\mathsf{max}} \geq \delta_{\mathsf{min}}$ and each time we find an acceptable step, we start the next iteration with a value of $\delta_+ \geq \delta_{\mathsf{min}}$. In short, δ_c can be reduced below δ_{min} while seeking an acceptable step, but $\delta_+ \geq \delta_{\mathsf{min}}$ 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* $\delta_{\mathsf{max}} \geq \delta_c \geq \delta_{\mathsf{min}} > 0$.

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: $\delta_+ = \max\{\delta_c, \delta_{\mathsf{min}}\}$.
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:

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

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

It is worth noting that in practice one might have another branch in which some $\eta_{\frac{1}{2}} \in (\eta_1, \eta_2)$ is used to reduce the trust-region radius if $\eta_1 \leq \frac{Ared_c}{Pred_c} \leq \eta_{\frac{1}{2}}$. A typical value for $\eta_{\frac{1}{2}}$ is .1, and the motivation is to try to avoid the expense of a next unacceptable trial step. 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, [7]. The present analysis would allow these niceties, but to avoid further complication, we do not include them here. Observe that in (5.5) and (5.6) we have expressed the quantities *Ared* and *Pred* as functions of ρ. Thus, although ρ_c does not effect the choice of the trial step s_c, we need to determine ρ_c *before* deciding the acceptance of 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 [16]. On the other hand,

global convergence theories developed by El-Alem [8], [9] and Powell and Yuan [23], require that the sequence $\{\rho_k\}$ be nondecreasing. El-Alem [8] 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 [9], since it ensures that the merit function is predicted to decrease 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 [33]. It can be stated as follows:

ALGORITHM 5.2. **Updating the penalty parameter**

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

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

   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*

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

   End if

The initial choice of the penalty parameter ρ_{-1} is arbitrary. However, it should be chosen consistent with the scale of the problem. Here, we take $\rho_{-1} = 1$ 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 δ_0, $\delta_{\min}$, $\delta_{\max}$, *and* $\varepsilon_{tol} > 0$.
Set $\rho_{-1} = 1$ *and* $\beta > 0$.

step 1. *(Test for convergence)*
If $\|W_c^T \nabla_x \ell(x_c)\| + \|C(x_c)\| \leq \varepsilon_{tol}$
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. *(This might be done by solving a trust-region subproblem since* $s_c^n = 0$ *is available. See Section 5.1)*
b) Set $s_c = s_c^t$.
else *(* $C(x_c) \neq 0$ *)*
a) Compute a quasi-normal step s_c^n *that satisfies a fraction of Cauchy decrease condition on the square norm 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. *(Perhaps not by solving a specific trust-region subproblem. See Section 5.1)*
End if
c) Set $s_c = s_c^n + s_c^t$.
End if
step 3. *(Update* λ_c*)*
Choose an estimate λ_+ *of the Lagrange multiplier vector.*
Set $\Delta\lambda_c = \lambda_+ - \lambda_c$.
step 4. *(Update the penalty parameter)*
Update ρ_- *to obtain* ρ_c *by using* **Algorithm 5.2.**
step 5. *(Evaluate the step)*
Compute

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

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 chosen to satisfy a sufficient predicted decrease condition, the fraction of Cauchy decrease. Note that in our algorithm, we assume that the tangential and the quasi-normal components of any trial step each satisfy this condition. In Lemma 7.2, 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 trial step length times

the penalty parameter. This is proved in Lemma 7.5. However, we emphasize again that the step is not chosen to maximize the predicted decrease.

We introduce some notation for the quantities computed during the trial steps. We have not introduced this notation up to now because it obscures the simplicity of the algorithm. However, in the analysis that follows we need to show some properties of every trial step, not just the successful steps $\{s_k\}$. Therefore, let δ_k^i, s_k^i, and ρ_k^i denote the quantities set by Algorithm 6.1 as it searches for an acceptable step. Thus, $\delta_k^0 = \delta_k$ at the first trial step of the kth iteration, s_k^0 is set by the first time though step 2, and ρ_k^0 is set using $\rho_k^{-1} = \rho_{k-1}$ the first time through step 4. If the trial step s_k^i is acceptable, then $s_k = s_k^i$, $\rho_k = \rho_k^i$, and δ_k^i is updated to become δ_{k+1}. In short, the algorithm is simpler to explain and code if one counts only successful steps. However, for the analysis, one needs a way to refer unambiguously to all the trial steps.

The model Lagrange multipliers also may depend on i. However, to keep the notation as simple as possible, we do not make this dependence explicit.

The penalty parameters ρ_k^i are shown to be bounded for $\epsilon_{tol} > 0$ as long as the algorithm does not terminate. The technique is to prove that, at any iteration k at which the penalty parameter is increased, we have: the product of the penalty parameter ρ_k^i and the trust-region radius δ_k^i is bounded by a constant that does not depend on k or i (this is done in Lemma 7.10); and the sequence of the trust-region radii δ_k^i is shown to be bounded away from zero (this is shown 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, under the assumption that the algorithm does not terminate, 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 at a given iterate, it either terminates, or finds an acceptable step after finitely many 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 8.4, it is shown that for any $\varepsilon_{tol} > 0$, the algorithm always terminates, *i. e.*, the termination condition of the algorithm will be met after finitely many iterations.

7.1. The problem assumptions. We start by stating the assumptions under which global convergence is proved for Algorithm 6.1. Assumptions A1 - A5 (see below) are used by Byrd, Schnabel, and Shultz [2], El-Alem [8], [9], [10] and Powell and Yuan [23] and their particular choices of Lagrange multiplier vectors satisfy A6.

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

A1. For all k, x_k and $x_k + s_k^i \in \Omega$, 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 uniformly bounded.

A6. The vectors $\lambda_k, k = 1, 2, ..$ are uniformly bounded.

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: $\|f(x)\| \leq \nu$, $\|\nabla f(x)\| \leq \nu_0$, $\|C(x)\| \leq \nu_1$, $\|\nabla C(x)\| \leq \nu_2$, $\|(\nabla C(x)^T \nabla C(x))^{-1}\| \leq \nu_3$, $\|\nabla^2 f(x)\| \leq \nu_4$, $\|\nabla^2 c_i(x)\| \leq \nu_5 \quad \forall\, i = 1, \cdots, m$, and $\|W(x)\| \leq \nu_6$.

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 $\|H_k\| \leq \nu_7$, $\|W_k^T H_k\| \leq \nu_7$, and

$\|W_k^T H_k W_k\| \leq \nu_7$.

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

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 condition (5.1) holds for the normal component $s_k^{i\,n}$ of s_k^i when it is truly normal to the tangent space.

LEMMA 7.1. *At the current iterate x_k, let the trial step component $s_k^{i\,n}$ actually be normal to the tangent space, then under the problem assumptions, there exists a constant $K_1 > 0$ independent of the iterates, such that*

$$\|s_k^{i\,n}\| \leq K_1\|C_k\|. \tag{7.1}$$

Proof. Because $s_k^{i\,n}$ is actually normal to the tangent space, we have

$$\begin{aligned}\|s_k^{i\,n}\| &= \|\nabla C_k(\nabla C_k^T \nabla C_k)^{-1}\nabla C_k^T s_k^i\| \\ &= \|\nabla C_k(\nabla C_k^T \nabla C_k)^{-1}(C_k + \nabla C_k^T s_k^i - C_k)\| \\ &\leq \|\nabla C_k(\nabla C_k^T \nabla C_k)^{-1}\|[\|C_k + \nabla C_k^T s_k^i\| + \|C_k\|] \ .\end{aligned}$$

Now, using the fact that $\|C_k + \nabla C_k^T s_k^i\| \leq \|C_k\|$, we have

$$\|s_k^{i\,n}\| \leq 2 \cdot \|\nabla C_k(\nabla C_k^T \nabla C_k)^{-1}\| \cdot \|C_k\| \ .$$

The rest follows from the problem assumptions. □

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 problem assumptions there exist positive constants K_2, K_3, and K_4 independent of the iterates such that*

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

and

$$\begin{aligned} q_k(s_k^{i\,n}) - q_k(s_k^i) \geq \\ \frac{\sigma}{2}\|W_k^T \nabla q_k(s_k^{i\,n})\| \min\{\frac{1-r}{\nu_6}\delta_k^i, K_4\|W_k^T \nabla q_k(s_k^{i\,n})\|\}. \end{aligned} \tag{7.3}$$

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

Now we deal with the trial steps assuming that they satisfy inequalities (7.2) and (7.3). In what follows, we will use implicitly that $\nabla C_k^T s_k^{i\,n} = \nabla C_k^T s_k^i$.

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

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

Proof. Consider

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

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

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

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 problem assumptions, there exist positive constants K_6, K_7 and K_8, independent of k, such that*

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

Proof. The proof follows directly from El-Alem [9]. □

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 problem assumptions, there exists a constant $K_9 > 0$ that does not depend on k, such that*

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

Proof. The proof follows directly from the above lemma and the fact that $\|s_k^i\|$ and $\|C_k\|$ are bounded. □

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

LEMMA 7.6. *Let s_k^i be generated by Algorithm 6.1. Then under the problem assumptions, for any positive ρ, the predicted decrease in the merit function satisfies*

$$\begin{aligned} Pred_k(s_k^i; \rho) \geq\ & \frac{\sigma}{2}\|W_k^T\nabla q_k(s_k^{i\,n})\| \min\{K_4\|W_k^T\nabla q_k(s_k^{i\,n})\|, \frac{1-r}{\nu_6}\delta_k^i\} \\ & - K_5\|C_k\| + \rho[\|C_k\|^2 - \|\nabla C_k^T s_k^i + C_k\|^2], \end{aligned} \tag{7.7}$$

where K_5 is as in Lemma 7.3.

Proof. We have

$$\begin{aligned} Pred_k(s_k^i; \rho) &= q_k(0) - q_k(s_k^i) - \Delta\lambda_k{}^T(C_k + \nabla C_k^T s_k^i) \\ &\quad + \rho[\|C_k\|^2 - \|\nabla C_k^T s_k^i + C_k\|^2] \\ &= (q_k(s_k^{i\,n}) - q_k(s_k^i)) \\ &\quad + (q_k(0) - q_k(s_k^{i\,n})) - \Delta\lambda_k{}^T(C_k + \nabla C_k^T s_k^i) \\ &\quad + \rho[\|C_k\|^2 - \|\nabla C_k^T s_k^i + C_k\|^2]. \end{aligned}$$

From (7.3) and Lemma 7.3, we have

$$\begin{aligned} Pred_k(s_k^i;\rho) &\geq \frac{\sigma}{2}\|W_k^T\nabla q_k(s_k^{i\,n})\| \min\{K_4\|W_k^T\nabla q_k(s_k^{i\,n})\| , \frac{1-r}{\nu_6}\delta_k^i\} \\ &\quad - K_5\|C_k\| + \rho[\|C_k\|^2 - \|\nabla C_k^T s_k^i + C_k\|^2]. \end{aligned}$$

Hence the result is established. □

If x_k is feasible, then the predicted reduction does not depend on ρ_k, so we take ρ_k 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* $\|C_k\| \leq \alpha\delta_k^i$ *where* α *satisfies:*

$$\alpha \leq \min\left\{ \frac{\varepsilon_{tol}}{3\delta_{\max}} , \frac{\varepsilon_{tol}}{3\nu_7 K_1 \delta_{\max}} , \frac{\sigma\varepsilon_{tol}}{12K_5}\min\{\frac{K_4\varepsilon_{tol}}{3\delta_{\max}} , \frac{1-r}{\nu_6}\}\right\} \tag{7.8}$$

then, for any positive ρ,

$$\begin{aligned} Pred_k(s_k^i;\rho) &\geq \frac{\sigma}{4}\|W_k^T\nabla q_k(s_k^{i\,n})\| \min\{K_4\|W_k^T\nabla q_k(s_k^{i\,n})\| , \frac{1-r}{\nu_6}\delta_k^i\} \\ &\quad + \rho[\|C_k\|^2 - \|\nabla C_k^T s_k^i + C_k\|^2]. \end{aligned} \tag{7.9}$$

Proof. If the algorithm does not terminate at x_k, then $\|W_k^T\nabla_x \ell_k\| + \|C_k\| > \varepsilon_{tol}$, and since $\|C_k\| \leq \alpha\delta_k^i$ with $\alpha \leq \frac{\varepsilon_{tol}}{3\delta_{\max}}$, therefore, $\|C_k\| \leq \frac{\varepsilon_{tol}}{3}$ and the reduced gradient satisfies $\|W_k^T\nabla_x \ell_k\| > \frac{2}{3}\varepsilon_{tol}$. Now,

$$\begin{aligned} \|W_k^T\nabla q_k(s_k^{i\,n})\| &= \|W_k^T(\nabla_x \ell_k + H_k s_k^{i\,n})\| \\ &\geq \|W_k^T\nabla_x \ell_k\| - \|W_k^T H_k s_k^{i\,n}\| \\ &\geq \frac{2}{3}\varepsilon_{tol} - \nu_7 K_1\|C_k\| \geq \frac{2}{3}\varepsilon_{tol} - \nu_7 K_1\alpha\delta_k^i. \end{aligned}$$

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

$$\|W_k^T\nabla q_k(s_k^{i\,n})\| \geq \frac{1}{3}\varepsilon_{tol}.$$

From Lemma 7.6, we have

$$\begin{aligned} Pred_k(s_k^i;\rho) &\geq \frac{\sigma}{2}\|W_k^T\nabla q_k(s_k^{i\,n})\| \min\{\frac{1-r}{\nu_6}\delta_k^i , K_4\|W_k^T\nabla q_k(s_k^{i\,n})\|\} \\ &\quad - K_5\|C_k\| + \rho[\|C_k\|^2 - \|\nabla C_k^T s_k^i + C_k\|^2]. \end{aligned}$$

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

$$\begin{aligned} Pred_k(s_k^i;\rho) &\geq \frac{\sigma}{4}\|W_k^T\nabla q_k(s_k^{i\,n})\| \min\{\frac{1-r}{\nu_6}\delta_k^i , K_4\|W_k^T\nabla q_k(s_k^{i\,n})\|\} \\ &\quad + \frac{\sigma}{12}\varepsilon_{tol}\min\left\{\frac{1-r}{\nu_6}\delta_k^i , \frac{\varepsilon_{tol}K_4}{3}\right\} \\ &\quad - K_5\alpha\delta_k^i + \rho[\|C_k\|^2 - \|\nabla C_k^T s_k^i + C_k\|^2]. \end{aligned}$$

Thus

$$\begin{aligned} Pred_k(s_k^i;\rho) \;\geq\; & \frac{\sigma}{4}\|W_k^T\nabla q_k(s_k^{i\,n})\| \min\{\frac{1-r}{\nu_6}\delta_k^i\,,\; K_4\|W_k^T\nabla q_k(s_k^{i\,n})\|\} \\ & + \frac{\sigma\varepsilon_{tol}\delta_k^i}{12}\min\left\{\frac{1-r}{\nu_6}\,,\;\frac{\varepsilon_{tol}K_4}{3\delta_{\max}}\right\} \\ & - K_5\alpha\delta_k^i + \rho[\|C_k\|^2 - \|\nabla C_k^T s_k^i + C_k\|^2], \end{aligned}$$

and since

$$\alpha \leq \frac{\sigma\varepsilon_{tol}}{12K_5}\min\left\{\frac{K_4\varepsilon_{tol}}{3\delta_{\max}}\,,\;\frac{1-r}{\nu_6}\right\},$$

we have

$$\begin{aligned} Pred_k(s_k^i;\rho) \;\geq\; & \frac{\sigma}{4}\|W_k^T\nabla q_k(s_k^{i\,n})\| \min\{K_4\,\|W_k^T\nabla q_k(s_k^{i\,n})\|\,,\;\frac{1-r}{\nu_6}\delta_k^i\} \\ & + \rho[\|C_k\|^2 - \|\nabla C_k^T s_k^i + C_k\|^2]. \end{aligned}$$

This completes the proof. □

Inequality (7.9) with $\rho = \rho_k^{i-i}$ guarantees that if the algorithm does not terminate and if $\|C_k\| \leq \alpha\delta_k^i$, then the penalty parameter at the current trial step 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 occur only when $\|C_k\| > \alpha\delta_k^i$.

LEMMA 7.8. *Given $\varepsilon_{tol} > 0$, there exists $K_{10} > 0$, which depends on ε_{tol}, but not on k or i, such that at any trial step s_k^i of iteration k at which the algorithm does not terminate and $\|C_k\| \leq \alpha\delta_k^i$ where α is as in Lemma 7.7, the following inequality holds*

$$Pred_k(s_k^i;\rho_k^i) \;\geq\; K_{10}\delta_k^i. \tag{7.10}$$

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

$$Pred_k(s_k^i;\rho_k^i) \geq \frac{\sigma\varepsilon_{tol}}{12}\min\left\{\frac{1-r}{\nu_6}\delta_k^i,\;\frac{K_4\varepsilon_{tol}}{3}\right\} \geq \frac{\sigma\varepsilon_{tol}}{12}\min\left\{\frac{1-r}{\nu_6},\;\frac{K_4\varepsilon_{tol}}{3\delta_{\max}}\right\}\delta_k^i.$$

Defining

$$K_{10} = \frac{\sigma\varepsilon_{tol}}{12}\min\left\{\frac{1-r}{\nu_6},\;\frac{K_4\varepsilon_{tol}}{3\delta_{\max}}\right\},$$

we have $Pred_k(s_k^i;\rho_k^i) \geq K_{10}\delta_k^i$ 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 $\{\delta_k^i\}$ of trust-region radii is bounded away from zero at those iterations for which the penalty parameter is increase at some trial step. This will allow us to conclude that the sequence $\{\rho_k^i\}$ of penalty parameters is bounded.

According to the rule for updating the penalty parameter, we use the penalty parameter from the previous trial step 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

$$Pred_k(s_k^i;\rho_k^{i-1}) \geq \frac{\rho_k^{i-1}}{2}[\|C_k\|^2 - \|C_k + \nabla C_k^T s_k^i\|^2], \tag{7.11}$$

then $\rho_k^i = \rho_k^{i-1}$. Otherwise, we use $\rho_k^i = \bar{\rho_k}^i + \beta$, which enforces (5.8). See Section 5.6.

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

1. *$\{\rho_k^i\}$ 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 (7.11) will hold.*

Proof. The proof is straightforward. □

LEMMA 7.10. *Let k, i be any pair of indices such that ρ_k^i is increased at the ith trial step of the kth iteration. If the algorithm does not terminate at x_k, then there exists $K_{11} > 0$ which depends on ε_{tol} but does not depend on k or i, such that for every $j \geq i$,*

$$\rho_k^j \delta_k^j \leq K_{11}. \tag{7.12}$$

Proof. If ρ_k^i is increased at the ith trial step of the kth iteration, then it is updated by the rule

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

Hence,

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

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:

$$\begin{aligned}
\frac{\rho_k^i K_2}{2}\|C_k\| \min \quad & \{ \quad r\delta_k^i \,,\, K_3\|C_k\| \,\} \\
&\leq -\frac{\sigma}{2}\|W_k^T \nabla q_k(s_k^{i\,n})\| \min \left\{ K_4\|W_k^T \nabla q_k(s_k^{i\,n})\| \,,\, \frac{1-r}{\nu_6}\delta_k^i \right\} \\
&\quad + K_5\|C_k\| - \beta(\nabla C_k C_k)^T s_k^{i\,n} - \frac{\beta}{2}\|\nabla C_k^T s_k^{i\,n}\|^2 \\
&\leq K_5\|C_k\| - \beta(\nabla C_k C_k)^T s_k^{i\,n} \\
&\leq K_5\|C_k\| + \beta\|\nabla C_k\| \, \|C_k\| \, \|s_k^{i\,n}\| \\
&\leq (K_5 + \beta\|\nabla C_k\| \, \|s_k^{i\,n}\|)\|C_k\|.
\end{aligned}$$

Then,

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

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

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

and

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

Now, if $j \geq i$, then $\delta_k^j \leq \delta_k^i$. Assume without loss of generality that $\rho_k^j = \rho_k^i$, i.e., that the ith trial step was the most recent increase with respect to j. Then $\rho_k^j\delta_k^j \leq \rho_k^i\delta_k^i$, and defining

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

we obtain the desired result. □

The following lemma gives a lower bound for the sequence $\{\delta_k^i\}$ for those iterates at which the algorithm does not terminate and the penalty parameter is increased. In the next section, we will be able to do away with the assumption that the penalty parameter is increased.

LEMMA 7.11. *Let the penalty parameter be increased at the ith trial step of the kth iteration. Then under the problem assumptions, if the algorithm does not terminate, there exists $\tilde{\delta}$, which depends on ε_{tol} but does not depend on the iterates, such that*

$$\delta_k^i \geq \tilde{\delta}. \tag{7.13}$$

Proof. To begin, we note that if $i = 0$, *i.e.* we are at the first trial step of iteration k, then by Algorithm 5.1, δ_k can not have gotten smaller than $\delta_{\min}$ during the course of the iteration. Thus, we can restrict our attention to the case where $i \geq 1$.

Our proof will consist in showing the existence of $\tilde{\delta}$ such that $\delta_k^i \geq \tilde{\delta}$ whether or not s_k^i is acceptable. Remember that for all the rejected trial steps we have $\delta_k^{j+1} = \alpha_1\|s_k^j\|$.

We consider two cases:

i) $\|C_k\| > \alpha\delta_k^j$ for all $j = 0, \ldots, i$.

ii) $\|C_k\| > \alpha\delta_k^j$ does not hold for some j between 0 and i.

i) Consider the case where the constraint violation $\|C_k\| > \alpha\delta_k^j$ for all $j = 0, \cdots, i$. We have from Lemma 7.5,

$$|Ared_k(s_k^j; \rho_k^j) - Pred_k(s_k^j; \rho_k^j)| \leq K_9\rho_k^j\|s_k^j\|^2.$$

Now since $\|C_k\| > \alpha\delta_k^j$, then from the way of updating ρ_k^j and using inequality (7.2), we have

$$\begin{aligned} Pred_k(s_k^j; \rho_k^j) &\geq \frac{\rho_k^j}{2}[\|C_k\|^2 - \|C_k + \nabla C_k^T s_k^j\|^2] \\ &\geq \frac{\rho_k^j}{2} K_2\|C_k\| \min\{K_3\alpha, r\}\delta_k^j. \end{aligned}$$

Hence

$$\frac{|Ared_k(s_k^j;\rho_k^j) - Pred_k(s_k^j;\rho_k^j)|}{Pred_k(s_k^j;\rho_k^j)} \leq \frac{2K_9\|s_k^j\|}{K_2\|C_k\|\min\{K_3\alpha,\, r\}}. \tag{7.14}$$

Since all the steps s_k^j for $j = 0, \cdots, i-1$ are rejected, it must be the case that

$$1 - \eta_1 < \left|\frac{Ared_k(s_k^j;\rho_k^j)}{Pred_k(s_k^j;\rho_k^j)} - 1\right|. \tag{7.15}$$

So from (7.14) and (7.15), we have

$$\|s_k^j\| \geq \frac{(1-\eta_1)K_2\min\{\alpha K_3,\, r\}}{2K_9}\|C_k\|, \quad \forall j = 0, \cdots, i-1. \tag{7.16}$$

Since $\delta_k^i = \alpha_1\|s_k^{i-1}\|$, and since $\|C_k\| > \alpha\delta_k^0$, it follows that

$$\delta_k^i = \alpha_1\|s_k^{i-1}\| \geq \alpha_1\left[\frac{(1-\eta_1)K_2\,\min\{\alpha K_3,\, r\}}{2K_9}\right]\alpha\delta_k^0. \tag{7.17}$$

Now, according to the rule for updating the trust-region radius, we know that $\delta_k^0 \geq \delta_{\min}$. Then

$$\delta_k^i \geq \frac{\alpha_1(1-\eta_1)K_2\min\{\alpha K_3,\, r\}}{2K_9}\alpha\delta_{\min} = K_{12}. \tag{7.18}$$

ii) If $\|C_k\| > \alpha\delta_k^j$ does not hold for all $j = 0, \cdots, i$, then there exists a largest index l, $0 \leq l < i$, such that $\|C_k\| \leq \alpha\delta_k^l$ holds.

If $i = l+1$ then, from the way of updating the trust-region radius, $\delta_k^i = \alpha_1\|s_k^l\|$. On the other hand, if $i \neq l+1$, since $\|C_k\| > \alpha\delta_k^j$, for all $j = l+1, \cdots, i$, then from (7.16) we have

$$\|s_k^j\| \geq \frac{(1-\eta_1)K_2\min\{\alpha K_3,\, r\}}{2K_9}\|C_k\|, \quad \forall j = l+1, \cdots, i-1.$$

Now, because s_k^{i-1} and s_k^{l+1} are rejected trial steps and using $\|C_k\| > \alpha\delta_k^{l+1}$, we can write

$$\begin{aligned}
\delta_k^i &= \alpha_1\|s_k^{i-1}\| \\
&\geq \alpha_1\frac{(1-\eta_1)K_2\min\{\alpha K_3,\, r\}}{2K_9}\|C_k\| \\
&\geq \alpha_1\alpha\frac{(1-\eta_1)K_2\min\{\alpha K_3,\, r\}}{2K_9}\delta_k^{l+1} \\
&\geq \alpha_1^2\alpha\frac{(1-\eta_1)K_2\min\{\alpha K_3,\, r\}}{2K_9}\|s_k^l\|. \qquad (7.19)
\end{aligned}$$

So, if we set

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

then we have

$$\delta_k^i \geq K_{13}\|s_k^l\|. \tag{7.20}$$

Therefore, using the above inequality and Lemma 7.10,

$$\rho_k^l \, \|s_k^l\| \leq \rho_k^i \frac{\delta_k^i}{K_{13}} \leq \frac{K_{11}}{K_{13}} = K_{14}.$$

From (7.5) we have

$$|Ared_k(s_k^l;\rho_k^l) - Pred_k(s_k^l;\rho_k^l)| \quad \leq \quad [K_6 + (K_7 + \alpha K_8)\rho_k^l\|s_k^l\|]\|s_k^l\|\delta_k^l.$$

Therefore,

$$|Ared_k(s_k^l;\rho_k^l) - Pred_k(s_k^l;\rho_k^l)| \quad \leq \quad [K_6 + (K_7 + \alpha K_8)K_{14}]\|s_k^l\|\delta_k^l. \tag{7.21}$$

Also, since $\|C_k\| \leq \alpha\delta_k^l$, then from Lemma 7.8, we have

$$Pred_k(s_k^l;\rho_k^l) \quad \geq \quad K_{10}\delta_k^l. \tag{7.22}$$

Using (7.21), (7.22) 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

$$\|s_k^l\| \quad \geq \quad \frac{(1-\eta_1)K_{10}}{K_6 + K_7K_{14} + \alpha K_8K_{14}}. \tag{7.23}$$

Now, using (7.20) and (7.23), we obtain the bound

$$\delta_k^i \geq K_{13}\frac{(1-\eta_1)K_{10}}{K_6 + K_7K_{14} + \alpha K_8K_{14}} = K_{15}.$$

Defining

$$\tilde{\delta} = \min\{\delta_{\min}, K_{12}, K_{15}\}$$

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 problem assumptions, if the algorithm does not terminate then there is some ρ^*, which depends on ε_{tol}, for which*

$$\lim_{k\to\infty} \rho_k \quad = \quad \rho^* < \infty. \tag{7.24}$$

Furthermore, there exists some index k_ρ such that $\rho_k = \rho^*$ for every $k \geq k_\rho$.

Proof. We need to show that $\rho^* \geq \rho_k^i$ for all pairs k, i. Clearly, it suffices to consider the sequence ρ_k^i of different ρ_k's where the double index k, i means that the penalty constant was increased to be ρ_k^i at the ith trial step of the kth iteration. Thus, there may be no terms or more than one term for a given k. Then from Lemma 7.10 and Lemma 7.11, we have

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

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

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

Now since the existence of ρ^* ensures that ρ_k is bounded, and since we know that when it is increased it is increased by at least β, there must be at most finitely many increases, and the proof is complete. □

This last result and the following one will play crucial roles in the proof of the global convergence of Algorithm 6.1.

LEMMA 7.13. *Under the problem assumptions, if the algorithm does not terminate then the augmented Lagrangian is bounded on* Ω

Proof. The proof is immediate from the boundedness of the penalty constant and the problem assumptions. □

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 under the assumption that it does not terminate. 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. This will allow us to drop the consideration of trial steps, and only consider "successful trial steps," $\{s_k\}$.

THEOREM 8.1. *Under the problem assumptions, unless some iterate* x_k *satisfies the termination condition of Algorithm 6.1, an acceptable step from* x_k *will be found after finitely many trial steps.*

Proof. The proof follows from Theorem 5.1 of El-Alem [9]. □

LEMMA 8.2. *Under the problem assumptions, assume that the algorithm does not terminate. Then there exists* $\delta_* > 0$, *which depends on* ε_{tol} *but does not depend on the iterates, such that for all* k, i,

$$\delta_k^i \geq \delta_*. \tag{8.1}$$

Proof. The proof is very similar to the proof of Lemma 7.11.

To begin, we note that if the first trial step is acceptable, then by Algorithm 5.1, δ_k can not have gotten smaller than $\delta_{\min}$ during the course of the iteration. Thus, we can restrict our attention to the case where there is at least one unsuccessful trial step. Let us assume then that we have j unsuccessful steps. Our proof will consist in showing the existence of $\tilde{\delta}$ such that $\delta_k^j \geq \tilde{\delta}$ whether or not s_k^j is acceptable, i.e., is s_k. Remember that for all the rejected trial steps we have $\delta_k^{j+1} = \alpha_1 \|s_k^j\| < \delta_k^j$.

We consider two cases:

i) $\|C_k\| > \alpha\delta_k^i$ for all $i = 0, \ldots, j$.

ii) $\|C_k\| > \alpha\delta_k^i$ does not hold for some i such that $0 < i \leq j$.

The proof of (i) is exactly the same as in the proof of Lemma 7.11, so let us proceed to (ii).

ii) Now if $\|C_k\| > \alpha\delta_k^i$ does not hold for all $i = 0, \ldots, j$. As in Lemma 7.11, we let l be the largest index such that $\|C_k\| \leq \alpha\delta_k^l$ holds. Now, since $\|C_k\| \leq \alpha\delta_k^i$ for all $i \leq l$,

it follows from Lemma 7.8 that for all such i, $Pred_k(s_k^i;\rho_k^i) \geq K_{10}\delta_k^i$. Furthermore, from Lemma 7.5, $|Ared_k(s_k^i;\rho_k^i) - Pred_k(s_k^i;\rho_k^i)| \leq K_9\rho_k^i\|s_k^i\|^2$, and because the step s_k^i is an unacceptable step, we have

$$1-\eta_1 < \left|\frac{Ared_k(s_k^i;\rho_k^i)}{Pred_k(s_k^i;\rho_k^i)} - 1\right| \leq \frac{K_9\rho_k^i\|s_k^i\|^2}{K_{10}\delta_k^i} \leq \frac{K_9\rho^*\|s_k^i\|}{K_{10}}.$$

The above inequality implies that, for all $i \leq l$,

$$\delta_k^i \geq \|s_k^i\| \geq \frac{(1-\eta_1)K_{10}}{K_9\rho^*}.$$

For all $i > l$, we have from (7.20) and the above inequality,

$$\delta_k^i \geq K_{13}\|s_k^l\| \geq K_{13}\frac{(1-\eta_1)K_{10}}{K_9\rho^*}.$$

It remains only to collect the constants as in Lemma 7.11. □

8.2. The global convergence results. Now we present our main global convergence result. Namely, under the problem 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). We start with a proof that if the algorithm does not terminate it will converge to a feasible point.

THEOREM 8.3. *Under the problem assumptions, if there exists $\varepsilon_{tol} > 0$, such that*

$$\|W_k^T\nabla_x\ell_k\| + \|C_k\| > \varepsilon_{tol}$$

for all k, then

$$\lim_{k\to\infty}\|C_k\| = 0. \tag{8.2}$$

Proof. We prove (8.2) by contradiction. We begin by assuming that there exists an infinite sequence of indices $\{k_j\}$ such that $\|C_k\|$ is bounded away from zero for all $k \in \{k_j\}$. This implies that there exists $\tau > 0$ such that for all $k \in \{k_j\}$, $\|C_k\| \geq \tau$. Now for each $k_j \geq k_\rho$ where k_ρ is as in Lemma 7.12, we have from (5.8) and (7.2) that

$$\begin{aligned} Pred_{k_j} &\geq \frac{\rho_{k_j}}{2}[\|C_{k_j}\|^2 - \|C_{k_j} + \nabla C_{k_j}^T s_{k_j}\|^2] \\ &\geq \frac{K_2\rho^*}{2}\|C_{k_j}\|\min\{K_3\|C_{k_j}\|, r\delta_{k_j}\} \\ &\geq \frac{K_2\rho^*\tau}{2}\min\{K_3\tau, r\delta_*\} = K_{16} > 0. \end{aligned}$$

Remember that we are only looking at successful steps at this point in the analysis so,

$$\mathcal{L}_{k_j} - \mathcal{L}_{k_j+1} = Ared_{k_j} \geq \eta_1 Pred_{k_j} \geq \eta_1 K_{16} > 0. \tag{8.3}$$

Since $\{\mathcal{L}_k\}$ is bounded below, a contradiction arises if we let k_j go to infinity. □

THEOREM 8.4. *Under the problem assumptions, given any $\varepsilon_{tol} > 0$, the algorithm terminates because*

$$\|W_k^T\nabla_x l_k\| + \|C_k\| < \varepsilon_{tol}. \tag{8.4}$$

Proof. Notice that if we suppose that the algorithm does not terminate and that some subsequence of $\{\|W_k^T \nabla_x \ell_k\|\}$ converges to zero, then nontermination is immediately contradicted by Theorem 8.3.

So, let us suppose that $\|W_k^T \nabla_x \ell_k\| \geq \tau_1$, for some $\tau_1 > 0$. Since $\|C_k\|$ goes to zero by Theorem 8.3 and the sequence of trust-region radii is bounded below by δ_*, there exists an index $N_1 > k_\rho$ such that for all $k \geq N_1$, $\|C_k\| \leq \alpha\delta_* \leq \alpha\delta_k$, with α as in (7.8). Therefore, by Lemma 7.8 with the i taken so that $s_k^i = s_k$ was the successful step, and by Lemma 8.2, we have again an infinite sequence of steps in which the actual decrease in $\mathcal{L}$ is at least $\eta_1 K_{10}\delta_*$. This contradicts the boundedness of $\mathcal{L}$ and 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 according 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 [20].

9.1. The Steihaug-Toint dogleg algorithm. This section is devoted to describing the generalized dogleg algorithm introduced by Steihaug [27] and Toint [30], 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, δ_c, *and* $\xi_c \leq \xi < 1$.

step 0: *(Initialization)*
Set $\hat{s}_0 = 0$.
Set $r_0 = -(G_c\hat{s}_0 + \nabla f_c)$.
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* $r_{i+1} = r_i - \alpha_i G_c d_i$.
If $\frac{\|r_{i+1}\|}{\|r_0\|} \leq \xi_c$, *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 quasi-normal component. We start our proposed step choice algorithm by finding a quasi-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 next iterate.

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 quasi-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 $\nabla C_c^T s + C_c = 0$ (see [5]). 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 Moore-Penrose pseudoinverse constraint normal and it requires no more than m iterations. Preconditioning is very important of course, but how to do 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 not difficult to prove that each Craig iterate is the ℓ_2 projection of the origin onto the subspace of the tangent space spanned by the steps up to that point and that each $\{s_j^{\mathsf{craig}}\}$ satisfies (5.1). Now, the Craig steps may not give monotone increasing ℓ_2 length, so a more agressive strategy that works perfectly well with our theory is to take the last pair of Craig iterates that straddle the trust-region boundary. In either case, by convexity, s_c^{dog} also satisfies (5.1). Furthermore, 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.

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 $\|s_c^{lf}\| \geq r\delta_c$, then we take

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

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

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

As long as $\{\|B_k^{-1}\|\}$ is uniformly bounded by some constant γ_*, s_c^{lf} satisfies (5.1) where the constant here is γ_*. This is a standard assumption for important classes of discretized optimal control problems, though it is stronger than our assumption that $[\nabla C(x_c)^T \nabla C(x_c)]^{-1}$ is uniformly bounded.

9.3. Computing the tangential component. We now assume that we have the quasi-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:

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

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 $\|W_c\bar{s}^t + s_c^n\| \leq \delta_c$.

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. **The CRG step choice algorithm**

Given $x_c \in \Re^n$, $\delta_c > 0$, *and* $\xi_c \leq \xi < 1$.

I. *FEASIBILITY:*

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

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

II. *MINIMIZATION:*

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

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

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

step 0: *(Initialization)*

Set $\hat{s}_0 = s_c^n$.

Set $r_0 = -W_c^T(H_c s_c^n + \nabla_x \ell_c)$.

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 $\hat{s}_{i+1} = \hat{s}_i + \alpha_i d_i$.

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* $r_{i+1} = r_i - \alpha_i W_c^T H_c d_i$.

If $\frac{\|r_{i+1}\|}{\|r_0\|} \leq \xi_c$, *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:**

It is worth noting here that this way of computing the tangent step does not have the property that once a step goes outside the trust region it could not come back in were the cg iteration continued. This means that the relaxed SQP step might lie inside the trust region, but the algorithm above might not return this more desirable step if the gradient scale and trust-region scale are inconsistent.

It would be better otherwise, of course, but the steps given here will lead to convergence, and we hope that near the solution when it becomes important to take SQP steps, the trust region will be large enough to compensate for the difference in shape. If the implementer wanted to be more agressive, there are various ways that fit our theory to deal with this situation. For example, we could take the dogleg step

based on the last time the cg iteration leaves the trust region rather than the first. Our concern here is to prove convergence theorems for the weakest conditions on the algorithm, and to show that reasonable algorithms satisfy those conditions, not to advocate particular implementation details of no consequence to the theory.

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 quasi-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 reducer matrix W_c. The proof is straightforward, so we will omit it.

LEMMA 9.3. *Under the problem assumptions, if there is a uniform bound on the matrix $B(x)^{-1}$, then the reducer matrix*

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

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

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

LEMMA 9.4. *Let s_c be a step generated by Algorithm 9.2 at the current iterate. Then s_c satisfies a fraction of Cauchy decrease condition on the quadratic model of the linearized constraints, i. e.,*

$$\|C_c\|^2 - \|C_c + \nabla C_c^T s_c\|^2 \quad \geq \quad K_2\|C_c\| \min\{r\delta_c \ , \ K_3\|C_c\|\}, \tag{9.1}$$

where K_2 and K_3 are constants independent of the iterates.

Proof. Suppose that we are applying Craig's 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

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

It is easy to see that

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

and

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

By convexity,

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

Thus,

$$\|C_c\|^2 - \|C_c + \nabla C_c^T s_c^{\mathbf{dog}}\|^2 \geq \|C_c\|^2 - \|C_c + \nabla C_c^T s_c^{cp}\|^2.$$

Thus we can apply Lemma 2.1.

Now suppose that s_c^n is given by $s_c^n = \gamma_c s_c^{\text{lf}}$ with $\gamma_c = \frac{r\delta_c}{\|s_c^{\text{lf}}\|}$ when $\|s_c^{\text{lf}}\| > r\delta_c$ and $\gamma_c = 1$ otherwise. When $\gamma_c = 1$, we have

$$\|C_c\|^2 - \|\nabla C_c^T s_c^n + C_c\|^2 \;=\; \|C_c\|^2 - \|\nabla C_c^T s_c^{\text{lf}} + C_c\|^2 = \|C_c\|^2.$$

When $\gamma_c < 1$, we have

$$\begin{aligned} \|C_c\|^2 - \|C_c + \nabla C_c^T s_c^n\|^2 &= \|C_c\|^2 - \|C_c + \gamma_c \nabla C_c^T s_c^{\text{lf}}\|^2 \\ &\geq \|C_c\|^2 - [\,(1-\gamma_c)\,\|C_c\| + \gamma_c\,\|C_c + \nabla C_c^T s_c^{\text{lf}}\|\,]^2 \\ &= [\,1 - (1-\gamma_c)^2\,]\,\|C_c\|^2 \geq \gamma_c \|C_c\|^2. \end{aligned}$$

The desired result will follow from the definition of s_c^{lf} and Lemma 9.3. □

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

$$q_c(s_c^n) - q_c(s_c) \;\geq\; \frac{\sigma}{2}\|W_c^T \nabla q_c(s_c^n)\| \min\left\{ K_4 \|W_c^T \nabla q_c(s_c^n)\|\,,\; \frac{(1-r)}{\nu_6}\delta_c \right\}.$$

Proof. Since we are solving the reduced problem

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

which is an unconstrained trust-region subproblem, the proof is immediate from Theorem 2.5 of Steihaug [27] followed by the use of the problem assumptions and Lemma 9.3. □

We state the following lemma here for completeness.

LEMMA 9.6. *The quasi-normal component computed by our proposed step choice algorithm satisfies*

$$\|s_c^n\| \leq K_1 \|C_c\|,$$

where K_1 is a positive constant independent of c.

Proof. The proof is given with the discussion of how to compute a quasi-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 [9] since it predicts

that the merit function is decreased at each iteration be at least a fraction of Cauchy decrease on the quadratic model of the linearized constraints. This indicates compatibility with the fraction of Cauchy decrease conditions imposed on the trial steps.

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 unspecified the way of approximating the Lagrange multiplier vector and the Hessian matrix.

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. This algorithm was one we had in mined as motivation for the convergence theory.

The least-squares or projection formula can be used as a scheme for estimating the multiplier since it fits the condition imposed on the multiplier updating scheme. Namely, under the standard assumptions, it produces bounded multipliers for the local models. For large problems, $\lambda = -B^{-1}\nabla_B f$ is likely to be a much preferable formula because of the cost of the least-squares solution. Furthermore, this will match better with the reducer matrix W, especially for problems where B can be easily identified. See Dennis and Lewis [6]. In either case, the uniform boundedness of $\{\lambda_k\}$ follows from the problem assumptions.

The exact Hessian matrix perhaps can be gotten by using automatic differentiation or an adjoint integration approach. 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 [29] 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 [6].

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, which we hope have been clarified. We thank Robert Michael Lewis and the referees for pointing out the importance of dealing with the quasi-normal component. We especially thank Luis Vicente for his careful and insightful reading.

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] E. CRAIG. The n-step iteration procedures. *J. Math. Physics*, 34:64–73, 1955.

[6] J. E. DENNIS Jr. and R. M. LEWIS. A comparison of nonlinear programming approaches to an elliptic inverse problem and a new domain decomposition approach. Technical report, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, 1994. in preparation.

[7] 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.

[8] 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.

[9] 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.

[10] M. M. EL-ALEM. A robust trust region algoritlun with non monotonic penalty parameter scheme for constrained optimization. Technical report, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, 1992. To appear in SIAM J. Opt.

[11] M. EL-HALLABI and R.A. TAPIA. A global convergence theory for arbitrary norm trust region methods for nonlinear equations. Technical report, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, TR93-41(replaces 87-25). Submitted for publication.

[12] R. FLETCHER. A class of methods for nonlinear programming with termination and convergence properties. In J. Abadie, editor, *Integer and Nonlinear Programming.* North-Holland, Amsterdam, 1970.

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

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

[15] 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.

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

[17] 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.

[18] 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.

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

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

[21] 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.

[22] 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.

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

[24] 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.

[25] 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.

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

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

[28] 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.

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

[30] 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.

[31] 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.

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

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

[34] J. ZHANG, N.-H. KIM, and L. LASDON. An improved successive linear programming algoritlun. *Management Science*, 31:1312–1331, 1985.