A TRUST-REGION APPROACH TO NONLINEAR SYSTEMS OF EQUALITIES AND INEQUALITIES*

J. E. DENNIS, JR.†, MAHMOUD EL-ALEM‡, AND KAREN WILLIAMSON§

Abstract. In this paper, two new trust-region algorithms for the numerical solution of systems of nonlinear equalities and inequalities are introduced. The formulation is free of arbitrary parameters and possesses sufficient smoothness to exploit the robustness of the trust-region approach. The proposed algorithms are one-sided least-squares trust-region algorithms. The first algorithm is a single-model algorithm, and the second one is a multimodel algorithm where the Cauchy point computation is a model selection procedure.

Global convergence analysis for the two algorithms is presented. Our analysis generalizes to nonlinear systems of equalities and inequalities the well-developed theory for nonlinear least-squares problems.

Numerical experiments on the two algorithms are also presented. The performance of the two algorithms is reported. The numerical results validate the effectiveness of our approach.

Key words. fraction of Cauchy decrease, global convergence, multimodel algorithm, nonlinear systems, nonlinear least squares, one-sided least squares, system of inequalities, trust-region methods, active-set strategies

AMS subject classifications. 65K05, 49D37

PII. S1052623494276208

1. Introduction. In this paper, we present two new trust-region algorithms for the numerical solution of a system of nonlinear equalities and inequalities defined by

\begin{align}
  c_i(x) &= 0, \quad i \in E, \\
  c_i(x) &\leq 0, \quad i \in I,
\end{align}

where $c_i : \mathbb{R}^n \to \mathbb{R}, I \cup E = \{1, \ldots, m\},$ and $I \cap E = \emptyset.$ In particular, we study trust-region methods for the following least-squares problem:

\begin{align}
  \min_{x \in \mathbb{R}^n} \frac{1}{2} \left\{ \sum_{i \in E} c_i(x)^2 + \sum_{i \in I} \left[ \max\{c_i(x), 0\} \right]^2 \right\}.
\end{align}

In practice, it is often useful to include weights on each term of the objective function (1.2), but here we omit them for simplicity.

Systems of nonlinear equalities and inequalities appear in a wide variety of problems in applied mathematics. These systems play a central role in the model formulation design and analysis of numerical techniques employed in solving problems arising in optimization, complementarity, and variational inequalities.

Best one-sided approximations have the form (1.2) (Taylor [35], Kaufman and Taylor [19], etc.). Another interest in problem (1.2) is when (1.1) is the constraint

*Received by the editors October 23, 1994; accepted for publication (in revised form) February 24, 1998; published electronically March 17, 1999. This research was supported by grants DOE DE-FG05-86ER25017, CRPC CCR-9120008, and AFOSR-F49620-9310212. http://www.siam.org/journals/siopt/9-2/27620.html
†Department of Computational and Applied Mathematics and Center for Research on Parallel Computation, Rice University, P.O. Box 1892, Houston, TX 77251 (dennis@rice.edu).
‡Department of Mathematics, Faculty of Science, Alexandria University, Alexandria, Egypt, and Center for Research on Parallel Computation, Rice University, P.O. Box 1892, Houston, TX 77251 (elalem@alex.eun.eg, mahmoud@rice.edu).
§TDA Research Inc., 12345 West 52nd Avenue, Wheat Ridge, CO 80033 (kaw@rice.edu).

291
set in a nonlinear programming problem. For this reason, it is important that we
design our algorithms to handle any relationship between \( m \) and \( n \), whereas, if we
were considering only the pure one-sided approximation generalization of nonlinear
least squares, then it would be reasonable to assume that \( m \geq n \). Also, the general
form of (1.2) we consider here is of interest in its own right (see Burke [5] and Burke
and Han [6]).

Newton’s method is a well-known and very powerful technique for solving non-
linear systems of equations. See, for example, Dennis and Schnabel [13]. Pshenichnyi
[31], Robinson [32], and Daniel [12] extended Newton’s method to nonlinear systems
of equalities and inequalities. Robinson [32] generalized Newton’s method to solve
problems in the form find \( x^* \) such that \( f(x^*) \in K \), where \( K \) is a nonempty closed

Burke and Han [6] considered a Gauss–Newton approach to solving generalized
inequalities; \( C(x) \leq_K 0 \), where \( C \) maps between normed linear spaces and \( \leq_K \) denotes
the partial order induced by the closed convex cone \( K \). Burke and Ferris [7] considered
an extension of the Gauss–Newton method to convex composite optimization. Using
tools from nondifferentiable optimization, they were able to establish a local quadratic
rate of convergence. By using a backtracking line search they were able to prove
global convergence. Many authors, including Garcia-Palomares and Restuccia [16],
Garcia-Palomares [15], and Burke [5], consider globally convergent algorithms for
solving problem (1.1). None of these theories are based on a trust-region globalization
strategy.

In this paper, we present two new trust-region-based algorithms for solving prob-
lem (1.1). By using an indicator matrix that will be presented in the next section,
we are able to transform our problem into one that possesses sufficient smoothness
to exploit the robustness of our algorithms. This allows us to use well-developed
tools and algorithms that require differentiability. In addition to that, the proposed
active set subproblems in this paper are much simpler than those proposed by Garcia-
Palomares and Restuccia [16] and Burke [5]. When we present our algorithms, it will
be clear that they are active set-type methods that try to identify the inequalities
likely to be violated at a solution to (1.2). Based on this property, we plan in future
research to use the ideas underpinning the algorithms developed here to develop an
\( \ell_2 \) trust-region active set algorithm for nonlinear programming.

The two algorithms we present in this paper are one-sided least-squares trust-
region algorithms. The first one is a single-model algorithm. The second one is
a multimodel algorithm, where the Cauchy point computation is a model selection
procedure. Most minimization algorithms use a local quadratic model where the
Hessian matrix may not be accurate. (See, e.g., Powell [30] and Toint [36].) Carter
[9], on the other hand, studied the case when the gradient might be inaccurate and
needs to be corrected during the step calculation. In this algorithm, we go all the
way to a model in which the function value may even be wrong, and it may need to
be corrected to find a step. See also Conn et al. [10].

We present global convergence results for the two algorithms. The two algorithms
were tested and compared on some test problems. The results are presented.

The rest of the paper is organized as follows. In section 2, we establish the
problem formulation and some notation. In section 3, we review the nonlinear least
squares problem and the concept of a fraction of the Cauchy decrease. In section 4,
we state some assumptions and prove a lemma that shows the required smoothness
properties of the problem formulation. In sections 5 and 6, we describe our two
trust-region algorithms. Section 7 is devoted to the global convergence theory for the first algorithm. In section 8, we present the global convergence theory for the multimodel algorithm. Section 9 contains our numerical results, and finally, we make some concluding remarks in section 10.

2. Preliminaries. It will be useful to establish some notation. Let $C(x) = (c_1(x), \ldots, c_m(x))^T$, and define the vector functions $C_E : \mathbb{R}^n \to \mathbb{R}^{|E|}$ to be the vector function whose components are $c_i(x)$ for $i \in E$ and $C_I : \mathbb{R}^n \to \mathbb{R}^{|I|}$ to be the vector function whose components are $c_i(x)$ for $i \in I$. Then, (1.1) can be written as

$$
C_E(x) = 0,
C_I(x) \leq 0.
$$

We define a $0$–$1$ diagonal indicator matrix $W(x) \in \mathbb{R}^{m \times m}$ whose diagonal entries are

$$
w_i(x) = \begin{cases} 1, & i \in E, \\ 1, & i \in I \text{ and } c_i(x) \geq 0, \\ 0, & i \in I \text{ and } c_i(x) < 0. 
\end{cases}
$$

(2.1)

It is also useful to identify the square submatrix $W_I(x)$ whose diagonal entries $w_i(x)$ correspond to $i \in I$. Now, we define the functions

$$
\Phi_E(x) = \frac{1}{2} C_E(x)^T C_E(x),
$$

(2.2)

$$
\Phi_I(x) = \frac{1}{2} C_I(x)^T W_I(x) C_I(x),
$$

(2.3)

and

$$
\Phi(x) = \Phi_E(x) + \Phi_I(x).
$$

(2.4)

The definition of $W(x)$ allows us to write $\Phi(x)$ as

$$
\Phi(x) = \frac{1}{2} C(x)^T W(x) C(x),
$$

and problem (1.2) can then be written as

$$
\min_{x \in \mathbb{R}^n} \Phi(x).
$$

It is easy to see that the function $t_+ = \max\{t, 0\}$ is continuous and that $(t_+)^2$ satisfies

$$
\frac{d}{dt} \left( \frac{1}{2} t_+^2 \right) = t_+.
$$

Hence, $(t_+)^2 \in C^1$. Thus, if each $c_i(x)$, for $i = 1, \ldots, m$, is continuously differentiable,

$$
\nabla \Phi(x) = \nabla \Phi_E(x) + \nabla \Phi_I(x)
= C'_E(x)^T C_E(x) + C'_I(x)^T W_I C_I(x)
$$

(2.5)
is well defined and continuous. This allows us to write
\[ \nabla \Phi(x) = C'(x)^T W(x) C(x). \]

Throughout the rest of the paper, the sequence of points generated by an algorithm will be denoted by \( \{ x_k \} \). Subscripted functions indicate that the function is evaluated at a particular point. For example, \( W_k \equiv W(x_k) \), \( C_k \equiv C(x_k) \), and so on. The expression \( f \in Lip(S) \) is used to mean that the function \( f \) is Lipschitz continuous at every point of the set \( S \). Finally, unless otherwise specified, all the norms used in this paper will be \( \ell_2 \)-norms.

3. Nonlinear least-squares trust-region algorithms. The nonlinear least-squares problem is traditionally written only for equalities as
\[ \min_{x \in \mathbb{R}^n} \Phi_E(x), \tag{3.1} \]
where \( \Phi_E(x) \) is given by (2.2).

A trust-region method for solving (3.1) is an iterative method that computes, at each iteration, a trial step \( s_k \) by minimizing a quadratic model of the objective function in the region in which we “trust” the model. First, we build a linear model \( C_E \) around the current iterate \( x_k \), namely, \( C_k(x_k) + C'_E(x_k) s \). Then we compute a trial step \( s_k \) that (approximately) solves the trust-region subproblem
\[
\begin{align*}
\min_{s \in \mathbb{R}^n} \quad & m_k(s) = \frac{1}{2} \| C'_E(x_k)s + C_E(x_k) \|^2 \\
\text{subject to} \quad & ||s|| \leq \Delta_k,
\end{align*}
\]
where \( \Delta_k > 0 \) is the radius of the trust region.

The trust-region approach was first suggested by Levenberg [20]. Later, Marquardt [21] used a different formulation of this technique, and the method is now known as the Levenberg–Marquardt method. More details about problem (3.1) and the trust-region subproblem can be found in Moré [22] and Dennis and Schnabel [14].

The global convergence analysis for problem (3.1) has been well established. To insure global convergence, the step can be required to satisfy a fraction of the Cauchy decrease condition. The Cauchy step minimizes the quadratic model along the negative gradient direction inside the trust region; i.e., \( s^\text{cp}_k = -\alpha^\text{cp}_k C'_E(x_k)^T C_E(x_k) \), where the step length is given by
\[ \alpha^\text{cp}_k = \begin{cases} \frac{\| C'_E(x_k)^T C_E(x_k) \|^2}{\| C'_E(x_k)^T C_E(x_k) \|^2 \| C'_E(x_k) + C_E(x_k) \|^2} & \text{if } \frac{\| C'_E(x_k)^T C_E(x_k) \|^2}{\| C'_E(x_k)^T C_E(x_k) \|^2 \| C'_E(x_k) + C_E(x_k) \|^2} \leq \Delta_k, \\
\frac{\| C'_E(x_k)^T C_E(x_k) \|^2}{\| C'_E(x_k)^T C_E(x_k) \|^2 \| C'_E(x_k)^T C_E(x_k) \|^2} & \text{otherwise.} \end{cases} \tag{3.2} \]

The fraction of the Cauchy decrease condition means that the step \( s_k \) must predict via the quadratic model of the function \( m_k(s) \) at least as much as a fraction of the decrease given by the Cauchy step \( s^\text{cp}_k \) on \( m_k(s) \); that is, there exists a constant \( \sigma > 0 \) fixed across all iterations, such that
\[ m_k(0) - m_k(s) \geq \sigma [m_k(0) - m_k(s^\text{cp})]. \tag{3.3} \]

Later, we will find it useful to work with the following condition instead of (3.3). If the step satisfies a fraction of Cauchy decrease, i.e., inequality (3.3), then
\[ m_k(0) - m_k(s_k) \geq \frac{\sigma}{2} \| C'_E(x_k)^T C_E(x_k) \| \min \left\{ \frac{\| C'_E(x_k)^T C_E(x_k) \|}{\| C'_E(x_k)^T C_E(x_k) \|}, \Delta_k \right\}. \tag{3.4} \]

More details can be found in Carter [8], Moré [23], and Powell [29].
4. **The continuity property.** Throughout this paper, we will require the following continuity and boundedness assumptions about the problem being solved.

**Assumption 1.** \( C'_E \) and \( C'_I \in \text{Lip}(\Omega) \), where \( \Omega \in \mathbb{R}^n \) is an open convex set.

**Assumption 2.** \( C'_E(x), \ C'_I(x), \ C'_E(x), \) and \( C'_I(x) \) are all bounded in norm for \( x \in \Omega \).

Equivalent to these assumptions is the existence of constants \( \gamma_E, \gamma_I \geq 0, \beta \geq \beta_i \geq 0 \) for \( i \in \{1, \ldots, m\} \), and \( b \geq 0 \), such that for all \( x, y \in \Omega \),

\[
\begin{align*}
(4.1) \quad & \|C'_E(x) - C'_E(y)\| \leq \gamma_E \|x - y\|, \\
(4.2) \quad & \|C'_I(x) - C'_I(y)\| \leq \gamma_I \|x - y\|, \\
(4.3) \quad & \|C'(x)\| \leq \beta, \\
(4.4) \quad & \|c'_i(x)\| \leq \beta_i, \quad i \in \{1, \ldots, m\}, \\
& \|C_E(x)\| \leq b, \quad \text{and} \\
& \|C_I(x)\| \leq b.
\end{align*}
\]

The following lemma establishes the Lipschitz continuity of \( \nabla \Phi(x) \) under Assumptions 1 and 2.

**Lemma 4.1.** *Let Assumptions 1 and 2 hold. Then, for every \( x, y \in \Omega \),

\[
(4.5) \quad \|\nabla \Phi(x) - \nabla \Phi(y)\| \leq a_0 \|x - y\|,
\]

where \( a_0 \) is a positive constant.*

*Proof. We have

\[
\|\nabla \Phi(x) - \nabla \Phi(y)\| \leq \|\nabla \Phi_E(x) - \nabla \Phi_E(y)\| + \|\nabla \Phi_I(x) - \nabla \Phi_I(y)\|.
\]

Now,

\[
\|\nabla \Phi_E(x) - \nabla \Phi_E(y)\| = \|C'_E(x)^T C_E(x) - C'_E(y)^T C_E(y)\|
\leq \left\| \left( C'_E(x) - C'_E(y) \right)^T C_E(x) \right\| + \|C'_E(y)^T (C_E(x) - C_E(y))\|
\leq \gamma_E \|x - y\| \cdot \|C_E(x)\| + \|C'_E(y)\| \cdot \|C_E(y) - C_E(x)\|
\leq \left( \gamma_E b + \beta_E \sum_{i \in E} \beta_i \right) \|x - y\|,
\]

where \( \beta_E \) bounds \( \|C'_E(y)\| \), which establishes the Lipschitz continuity of \( \nabla \Phi_E(x) \).

Also, we can bound \( \|C'_I(y)\| \) by \( \beta_I \). Hence, using an argument similar to what we used in (4.6), we have

\[
\|\nabla \Phi_I(x) - \nabla \Phi_I(y)\| \leq \beta_I \|x - y\| + \beta_I \|W_I(y)C_I(y) - W_I(x)C_I(x)\|.
\]

We will complete the proof by showing that \( W_I(\cdot)C_I(\cdot) \) is Lipschitz continuous. Consider a fixed \( i \in I \), and let \( Z_i = \{z \in \mathbb{R}^n : c_i(z) = 0\} \). If \( c_i(x) \cdot c_i(y) < 0 \), then we can choose \( z_i \in Z_i \cap [x, y] \). Thus,

\[
w_i(x)c_i(x) - w_i(y)c_i(y) = \begin{cases} c_i(x) - c_i(z_i) & \text{if } c_i(y) < 0, \\
 c_i(z_i) - c_i(y) & \text{if } c_i(y) > 0,
\end{cases}
\]

and so by (4.4),

\[
|w_i(x)c_i(x) - w_i(y)c_i(y)| \leq \beta_i \max \{\|x - z_i\|, \|z_i - y\|\} \leq \beta_i \|x - y\|.
\]
If \( c_i(x) \cdot c_i(y) \geq 0 \), then for \( c_i(y) \neq 0 \),

\[
\begin{align*}
  w_i(x)c_i(x) - w_i(y)c_i(y) &= \begin{cases} 
  0 & \text{if } c_i(y) < 0, \\
  c_i(x) - c_i(y) & \text{if } c_i(y) > 0,
  \end{cases}
\end{align*}
\]

and so \( |w_i(x)c_i(y)| \leq \beta_i \|x - y\| \). If \( c_i(y) = 0 \), then similarly, \( |w_i(x)c_i(y)| \leq \beta_i \|x - y\| \).

Putting the components together yields

\[
\|W_I(y)C_I(y) - W_I(x)C_I(x)\| \leq \left( \sum_{i \in I} \beta_i \right) \|x - y\|.
\]

Finally, we obtain

\[
\|\nabla \Phi_I(x) - \nabla \Phi_I(y)\| \leq \left( \gamma_I b + \beta_I \left( \sum_{i \in I} \beta_i \right) \right) \|x - y\|.
\]

This completes the proof. 

5. Single-model algorithm. We describe the single-model algorithm for (1.2) in four sections. In section 5.1, we discuss the model and the trust-region subproblem, and section 5.2 describes the method of solving this subproblem. Section 5.3 is devoted to presenting the trial-step acceptance mechanism and the trust-region updating strategy. Finally, in section 5.4 we summarize the single-model algorithm.

5.1. The trust-region subproblem. The idea here is to use a standard trust-region algorithm for nonlinear least squares on

\[
\min_{x \in \mathbb{R}^n} \Phi(x),
\]

where \( \Phi(x) \) is given by (2.4).

At the current iterate \( x_k \), the set of binding or violated inequalities at \( x_k \) is identified, and the 0–1 diagonal matrix \( W(x_k) \) defined by (2.1) is assembled. Next, we build a quadratic model

\[
q(x_k + s) = \frac{1}{2} \|W(x_k)(C'(x_k)s + C(x_k))\|^2
\]

of \( \Phi \) around the current iterate \( x_k \), as in the Gauss–Newton approach, where

\[
\Phi(x_k + s) = \frac{1}{2}C(x_k + s)^T W(x_k + s)C(x_k + s).
\]

Thus, the quadratic model contains information about only those inequalities which are violated or active at \( x_k \).

At each iteration, a trial step \( s_k \) is computed as an approximate solution to the trust-region subproblem

\[
\begin{align*}
  \minimize q(x_k + s) &\equiv \frac{1}{2} \|W(x_k)(C'(x_k)s + C(x_k))\|^2 \\
  \text{subject to } &\|s\| \leq \Delta_k
\end{align*}
\]

where \( \Delta_k > 0 \) is the current trust-region radius.

5.2. Solution of the single-model subproblem. We want to compute an approximate solution to the trust-region subproblem (5.2). The most obvious strategy would be to use an algorithm such as the one used in the MINPACK routine LMDER [24], which is based on the Levenberg–Marquardt approach [22]. However, this routine will not solve underdetermined systems; i.e., it requires that the dimension of \( C \) be
at least as large as the dimension of $x$. In the context of finding a solution to a set of equalities and inequalities, we have not assumed any relationship between $n$ and $m$. Another alternative to consider is the completely general Moré–Sorensen routine GQTPAR [1]. However, this routine is unsuitable for $n < m$ because of its strategy for the hard case. (See Moré and Sorensen [25] for the definition of the hard case.) This routine always steps to the boundary of the trust region even when a zero-residual step is safely inside. We emphasize that this is not a criticism of GQTPAR; it is a statement of our special needs. Hence, we will use a dogleg algorithm to solve the subproblem (5.2).

The dogleg algorithm approximates the solution curve to problem (5.2) by a piecewise linear function connecting the Cauchy point to the “Newton” point. The Cauchy step is defined to be $s_k^\text{cp} = -\alpha_k^\text{cp} C'(x_k)^T W_k C(x_k)$, where

$$
\alpha_k^\text{cp} = \begin{cases} 
\frac{\|C'(x_k)^T W_k C(x_k)\|^2}{\|W_k C'(x_k)^T W_k C(x_k)\|} & \text{if } \frac{\|C'(x_k)^T W_k C(x_k)\|^3}{\|C'(x_k)^T W_k C(x_k)\|^2} \leq \Delta_k, \\
\frac{3}{\|C'(x_k)^T W_k C(x_k)\|} & \text{otherwise}.
\end{cases}
$$

If $s_k^\text{cp}$ lies inside the trust region, then we compute the step that will play the role of the Newton step, and it is the minimum norm solution to

$$
(5.3) \quad \text{minimize } \frac{1}{2} \|W(x_k) (C'(x_k)s + C(x_k))\|^2.
$$

We will refer to this step as $s_k^\text{lf}$. To compute this step, one can use the routine GELSX from LAPACK [1], which can handle both over- and underdetermined systems.

When $n$ is large, iterative methods might have to be used to obtain the minimum norm solution of problem (5.3). A truncation procedure might also be needed. For example, a Steihaug–Toint-type algorithm can be used [34], [37].

The algorithm described in Golub and von Matt [17] is also of interest, especially for the large-scale case. This algorithm is applicable to the over- and underdetermined cases and can be applied directly to solve the trust-region subproblem (5.2).

Using the minimum norm solution ensures that if the computed step is outside the trust region, then there are no other solutions to (5.3) that are inside the trust region. If $s_k^\text{lf}$ is inside the trust region, then we take it as the solution to the subproblem. Otherwise, we compute the dogleg step between the Cauchy point and $s_k^\text{lf}$ with length $\Delta_k$, and take it as the trial step.

**Algorithm 5.1.** Computing a Trial Step.

**Compute the Cauchy step,** $s_k^\text{cp} = -\alpha_k^\text{cp} (C_k')^T W_k C_k$.

If ($\|s_k^\text{cp}\| = \Delta_k$), then set $s_k = s_k^\text{cp}$.

Else, if ($\nabla q_k(s_k^\text{cp}) = 0$), then set $s_k = s_k^\text{cp}$.

Else, compute $s_k^\text{lf}$, the minimum norm solution to

$$
\text{minimize } \frac{1}{2} \| W_k C_k + W_k (C_k')^T s \|^2
$$

If ($\|s_k^\text{lf}\| \leq \Delta_k$), then set $s_k = s_k^\text{lf}$.

Else, dogleg between $s_k^\text{cp}$ and $s_k^\text{lf}$.

Since our theory is based on the fraction of the Cauchy decrease condition (see section 3), any method that computes a trial step that gives at least a fraction of the Cauchy decrease can be used. Therefore, in the case when $n$ is large, a generalized dogleg algorithm introduced by Steihaug [34] and Toint [37] can be used to compute the trial step $s_k$. This algorithm is based on the linear conjugate gradient method and is known to be suitable for large problems for which effective preconditioners are known.
5.3. Accepting the step and updating the trust-region. Once we have computed a trial step \( s_k \), we decide if the step is acceptable by comparing the amount of reduction \( s_k \) predicts in the model (5.1) to the amount of reduction we actually obtain in \( \Phi(x) \). The actual reduction in \( \Phi(x) \) is given by

\[
\text{Ared}_k = \Phi(x_k) - \Phi(x_k + s_k)
\]

\[
= \frac{1}{2} \| W_k C_k \|^2 - \frac{1}{2} \| W(x_k + s_k) C(x_k + s_k) \|^2;
\]

and the predicted reduction in the model is given by

\[
\text{Pred}_k = q(x_k) - q(x_k + s_k)
\]

\[
= \frac{1}{2} \| W_k C_k \|^2 - \frac{1}{2} \| W_k (C_k s_k + C_k) \|^2.
\]

The trust-region algorithm should produce steps that decrease \( \Phi \) and make progress toward the feasible region. To guarantee this, the actual reduction in \( \Phi \) has to be greater than some fraction of the predicted reduction in the model for the step to be deemed acceptable. The computation of the trial step (specifically, the solution of the trust-region subproblem, discussed in section 5.2) ensures that the step predicts at least a fraction of the Cauchy decrease in the model, and hence, \( \text{Pred}_k > 0 \).

The step is accepted if \( \eta_1 \leq \frac{\text{Ared}_k}{\text{Pred}_k} \), where \( \eta_1 \in (0, 1) \) is a small fixed constant, say \( 10^{-4} \). If the step is judged acceptable, then we proceed to the next iteration. Otherwise, the trial step is rejected, the trust-region radius is decreased by setting \( \Delta_k = \alpha_1 \| s_k \| \) for some \( \alpha_1 \in (0, 1) \), and another trial step is computed from \( x_k \) in the smaller trust region.

If the step is accepted, then the trust-region radius is updated by comparing the value of \( \text{Ared}_k \) with \( \text{Pred}_k \). Namely, if \( \eta_2 \leq \frac{\text{Ared}_k}{\text{Pred}_k} < \eta_3 \), where \( \eta_2 \in (\eta_1, 1) \), then the radius of the trust region is kept the same. If the agreement between the actual reduction and the predicted reduction is poor (\( \frac{\text{Ared}_k}{\text{Pred}_k} < \eta_2 \), where \( \eta_2 \) is less than or equal to 0.1), then we allow possible reduction in the radius of the trust region. We set \( \Delta_{k+1} = \min(\Delta_k, \alpha_2 \| s_k \|) \) for some \( \alpha_2 \geq 1 \). If, on the other hand, the agreement between the actual reduction and the predicted reduction is fair, \( \eta_3 \leq \frac{\text{Ared}_k}{\text{Pred}_k} < \eta_4 \), possibly increase the trust region. Set \( \Delta_{k+1} = \max(\Delta_k, \alpha_2 \| s_k \|) \). Typical values for \( \eta_3 \) and \( \eta_4 \) are 0.25 and 0.75. If the agreement between the actual reduction and the predicted reduction is good, \( \eta_4 \leq \frac{\text{Ared}_k}{\text{Pred}_k} \), then increase the trust region radius. Set \( \Delta_{k+1} = \max(\alpha_2 \Delta_k, \alpha_3 \| s_k \|) \), where \( \alpha_3 \geq \alpha_2 \).

Further details concerning the basic trust-region framework can be found, for instance, in Moré [23] or Dennis and Schnabel [14].

5.4. Summary of the single-model algorithm. Putting the pieces together, we can now outline the single-model algorithm for finding a local minimizer of \( \Phi(x) \).

**Algorithm 5.2.** The Single-Model Algorithm.

**Initialization:** Given \( x_0 \) and \( \Delta_0 > 0 \), compute \( C(x_0) \), \( W(x_0) \), and \( C'(x_0) \).

**At every iteration, do**

1. **Step 1.** Check for convergence.
2. **Step 2.** Compute a trial step \( s_k \) by approximately solving

   minimize \( q(x_k + s) = \frac{1}{2} \| W_k (C_k s + C_k) \|^2 \)

   subject to \( \| s \| \leq \Delta_k \).
Step 3. Compute $C(x_k + s_k)$ and $W(x_k + s_k)$.
Step 4. Decide if the step is acceptable based on the ratio of $A_{red_k}$ to $Pred_k$ (as discussed in section 5.3) and update $\Delta_k$ (by the mechanism given in section 5.3).
Step 5. If the step $s_k$ is acceptable, set $x_{k+1} = x_k + s_k$, compute $C'(x_{k+1})$, and go to Step 1.
Else set $k := k + 1$ and go to Step 2.

6. Multimodel algorithm. We describe our multimodel algorithm in four sections. In sections 6.1–6.3, we describe our way of computing the trial step. Section 6.4 is devoted to presenting the trial-step acceptance mechanism and how to update the trust-region radius.

6.1. The trust-region subproblem. At each iteration $k$, the set of binding or violated inequalities at $x_k$ is identified and the 0–1 diagonal matrix $W(x_k)$, defined by (2.1), is assembled.
Next, a generalized Cauchy point is computed to be the point $x_k + s_{gcp}$ where $s_{gcp}$ solves the one-dimensional piecewise-quadratic convex minimization problem
\[
\text{minimize } \frac{1}{2} \| V_k(s)(C'_k s + C_k) \|^2 \\
\text{subject to } \| s \| \leq \Delta_k, \ s = -\alpha C'_k T W_k C_k,
\]
where $V_k(s)$ is another 0–1 diagonal indicator matrix, whose diagonal elements $v_i$ are
\[
v_i(s) = \begin{cases} 
1, & i \in E, \\
1, & i \in I, \ w_i(x_k) = 1, \text{ and } c_i(x_k) + c'_i(x_k) s \geq 0, \\
0, & \text{otherwise.}
\end{cases}
\]

The algorithm then computes a trial step by solving the following standard trust-region subproblem:
\[
\text{minimize } \frac{1}{2} \| V_k(s_{gcp})(C'_k s + C_k) \|^2 \\
\text{subject to } \| s_k \| \leq \Delta_k.
\]

It will be useful later if we point out here that
\[
\Psi_k(s) = \frac{1}{2} \| V_k(s)(C'_k s + C_k) \|^2
\]
is a local form of $\Phi$ for the linearization of $C$ about $x_k$. It is easy to see that $\Psi_k(0) = \Phi(x_k), \nabla_s \Psi_k(s) = C'_k T V_k(s)(C'_k s + C_k)$, and $\nabla_s \Psi_k(0) = C'_k T W_k C_k$.

The mappings $W_k$ and $V_k$ are very reminiscent of the structure functionals used by Osborne and Womersley. Interested readers are referred to [26] and [27].

6.2. Computing the generalized Cauchy point. In this section, we present the algorithm that we use to compute the generalized Cauchy point and thus to determine which inequalities will be included in the trial step calculation. At the $k$th iteration, given $C(x_k)$, we form the nonlinear indicator matrix $W(x_k)$, given by (2.1).
The generalized Cauchy point $s^{\text{gcp}}$ solves the trust-region subproblem

$$
\begin{align*}
\text{minimize} & \quad q_k(s) = \frac{1}{2} \| V_k(s) (C_k + C_k' s) \|^2 \\
\text{subject to} & \quad s = -\alpha (C_k')^T W_k C_k, \\
& \quad \| s \| \leq \Delta_k,
\end{align*}
$$

(6.3)

where $V_k(s)$ is a diagonal 0–1 indicator matrix that designates which of the linearized inequalities that are binding or violated at $s = 0$ are still violated or binding for a particular step $s$. As with $W(x)$, the diagonal elements corresponding to equalities are always one, and $V_k(s)$ is given by (6.1).

In problem (6.3), the step is restricted to the negative gradient direction of the model at $x_k$. Normalizing this direction makes the trust-region constraint invariant under the resulting change of variables, and so we will refer to

$$
d_{\text{cp}}^k = -((C_k')^T W_k C_k) \| (C_k')^T W_k C_k \|
$$

as the Cauchy direction.

It is easy to see that $V_k(s)$ evaluated at $s = 0$ is just $W_k$. Furthermore, the only components of $V_k(s)$ that actually depend on $s$ are those $v_i$ corresponding to inequalities that are binding or violated at $x_k$. Consequently, if all of the inequalities are strictly satisfied at $x_k$ (or if there are no inequalities), then $V(s) \equiv W_k$ for all $s$. In this case, $V_k(s^{\text{gcp}}) = W_k$, and problem (6.3) reduces to the standard Cauchy point computation with $s^{\text{gcp}} = \alpha d_{\text{cp}}^k$, where

$$
\alpha_k = \min \left\{ \frac{\| (C_k')^T W_k C_k \|}{\| W_k C_k' d_{\text{cp}}^k \|^2}, \Delta_k \right\}.
$$

(6.4)

For the remainder of this section, we will assume that there is at least one violated or binding inequality at $x_k$. Thus, we have a one-dimensional trust-region subproblem of the form

$$
\begin{align*}
\text{minimize} & \quad q_k(\alpha) = \frac{1}{2} \| V_k(\alpha) (C_k + \alpha C_k' d_{\text{cp}}^k) \|^2 \\
\text{subject to} & \quad |\alpha| \leq \Delta_k.
\end{align*}
$$

(6.5)

Note that we obtain $V_k$ as a function of $\alpha$ alone through the substitution $s = \alpha d_{\text{cp}}^k$. The objective function of problem (6.5) is a one-dimensional piecewise quadratic, and it is convex and continuously differentiable.

It is worth noting that the number of ones in $V_k(\alpha)$ decreases as $\alpha$ increases and that for $\alpha_1 \leq \alpha_2$, the following inequality holds for all $\alpha$:

$$
\| V_k(\alpha_1) (C_k + \alpha C_k' d_{\text{cp}}^k) \| \geq \| V_k(\alpha_2) (C_k + \alpha C_k' d_{\text{cp}}^k) \|.
$$

We will use a “piecewise” form of Newton’s method to solve problem (6.5). Starting from $\alpha_0 = 0$, we fix the indicator matrix at $V_k(\alpha_j)$ and form the $j$th quadratic model

$$
q_j^k(\alpha) = \frac{1}{2} \| V_k(\alpha_j) (C_k + \alpha C_k' d_{\text{cp}}^k) \|^2.
$$

(6.6)
The $j$th quadratic model $q_k^j(\alpha)$ is equal to the composite quadratic model $q_k(\alpha)$ given in (6.6) at the point $\alpha_j$. Then, we minimize this model subject to the trust-region constraint to obtain the next iterate $\alpha_{j+1}$. The Newton step for (6.6) is

\begin{equation}
\alpha_{j+1} = -\frac{C_k^T V_k(\alpha_j) C_k \delta_k^p}{\|V_k(\alpha_j) C_k \delta_k^p\|^2}.
\end{equation}

Since the quadratic model is one-dimensional and convex, either the Newton step is the solution to the trust-region subproblem or the trust-region is binding, in which case $\alpha_{j+1} = \Delta_k$. Given the new iterate, $\alpha_{j+1}$, we must determine if it is the solution to (6.5). First, we evaluate $V_k(\alpha_{j+1})$ to determine if the set of linearly violated or binding inequalities has changed. The following lemma gives sufficient conditions for $\alpha_{j+1}$ to solve (6.5).

**Lemma 6.1.** Let $\alpha_{j+1}$ minimize the $j$th quadratic model $q_k^j(\alpha)$ inside the trust region. Then, $\alpha_{j+1}$ solves the trust-region subproblem (6.5) if one of the following conditions holds:

i. $V_k(\alpha_{j+1}) = V_k(\alpha_j)$.

ii. The trust-region is binding at $\alpha_{j+1}$.

iii. The gradient of the new quadratic model,

\begin{equation}
\nabla q_k^{j+1}(\alpha_{j+1}) = (C_k + \alpha_{j+1} C_k' C_k \delta_k^p) V_k(\alpha_{j+1}) C_k' \delta_k^p,
\end{equation}

is equal to zero.

Furthermore, the algorithm must terminate with $\alpha_{j+1}$ that satisfies either i, ii, or iii in a finite number of iterations.

**Proof.** i. Since $V_k(\alpha_{j+1}) = V_k(\alpha_j)$, we have

\begin{align*}
q_k^j(\alpha_{j+1}) &= \frac{1}{2} \|V_k(\alpha_j) (C_k + \alpha_{j+1} C_k' \delta_k^p)\|^2 \\
&= \frac{1}{2} \|V_k(\alpha_{j+1}) (C_k + \alpha_{j+1} C_k' \delta_k^p)\|^2 = q_k^{j+1}(\alpha_{j+1}).
\end{align*}

Since $q_k^{j+1}(\alpha_{j+1})$ is the composite quadratic $q_k$ at $\alpha_{j+1}$, $\alpha_{j+1}$ solves (6.5).

ii. Now assume that $V_k(\alpha_{j+1}) \neq V_k(\alpha_j)$. Then from the definition of $V_k$ the only possibility is that at least one of the linear inequalities that was violated or binding at $\alpha_j$ is strictly satisfied at $\alpha_{j+1}$. Without loss of generality, we will assume that there is only one such inequality with index $l$ such that $V_{lk}(\alpha_j) = 1$ and $V_{lk}(\alpha_{j+1}) = 0$.

We need to show that if $\alpha_{j+1} = \Delta_k$ minimizes $q_k^j(\alpha)$ subject to the trust-region constraint, then it also minimizes $q_k^{j+1}(\alpha)$ in the trust region. Thus, $\alpha_{j+1}$ will be a solution to (6.5) because $q_k^{j+1}(\alpha_{j+1})$ is the composite quadratic model at $\alpha_{j+1}$.

The only difference between $q_k^j(\alpha_{j+1})$ and $q_k^{j+1}(\alpha_{j+1})$ is in the $l$th term, with $V_{lk}(\alpha_j) = 1$ and $V_{lk}(\alpha_{j+1}) = 0$. So,

\begin{equation}
\nabla q_k^j(\alpha_{j+1}) = \nabla q_k^{j+1}(\alpha_{j+1}) + (c_{lk} + \alpha_{j+1} c_{lk}' \delta_k^p) c_{lk}' \delta_k^p.
\end{equation}

Since $V_{lk}(\alpha_j) = 1$, the definition of $V_k$ indicates that $W_l(x_k) = 1$, which implies that $c_{lk} \geq 0$. Also, $V_{lk}(\alpha_{j+1}) = 0$ yields $c_{lk} + \alpha_{j+1} c_{lk}' \delta_k^p < 0$.

We can conclude that $\alpha_{j+1} \in [0, \Delta_k]$ from the fact that $\nabla q_k(0) \leq 0$ because either $\delta_k^p = 0$ and $\alpha = 0$ solves (6.5) or $\alpha \delta_k^p$ is a descent direction for $q_k$ at $\alpha = 0$, i.e.,

\begin{equation}
\nabla q_k(0) = C_k^T W_k C_k \delta_k^p = -\|(C_k^T W_k C_k)\| \leq 0.
\end{equation}
Thus, $\alpha_{j+1} \geq 0$, and so $c''_k d_k \leq 0$. From this we obtain

$$
(c_k + \alpha_{j+1} c''_k) c'_k d_k \geq 0.
$$

(6.10)

In other words, one previously violated linear inequality became strictly satisfied or binding on the interval $[\alpha_j, \alpha_{j+1}]$.

For $\alpha_{j+1}$ to minimize $q^j_k(\alpha)$ with the trust-region constraint binding, we know that

$$
\nabla \alpha q^j_k(\alpha_{j+1}) \leq 0.
$$

Combining this with (6.9) and (6.10), we can conclude that $\nabla q^j_k(\alpha_{j+1}) \leq 0$, and since the trust region is binding, $\alpha_{j+1}$ solves (6.5).

iii. If $\nabla q^j_k(\alpha_{j+1}) = 0$, then clearly $\alpha_{j+1}$ solves (6.5).

Finally, the algorithm must terminate in at most $\|W_k(x_k)\|^2 + 1$ iterations. At every iteration, at least one linear inequality, with $w_i(x_k) = 1$, must become strictly satisfied, or $i$ indicates that the algorithm will terminate at that iteration. We started with $\|W_k(x_k)\|^2$ violated or binding linear inequalities, so by iteration $\|W_k(x_k)\|^2$, either we have found the solution or all of the inequalities with $w_i = 1$ now have $v_{ik}(\alpha) = 0$. Then, only the equalities, $i \in E$, have $v_{ik} = 1$, and one more iteration could be required to solve (6.5).

Now after Lemma 6.1 has established constructive stopping criteria, we can state our algorithm for computing the generalized Cauchy point.

Algorithm 6.2. Generalized Cauchy Point Algorithm.

Initialization.

Given $x_k$, $C_k$, $W_k$, $C'_k$, and $\Delta_k > 0$.

Set $j = 0$, $\alpha_0 = 0$, and $V_0 = W_k$.

Compute $s_k^{(CP)}$ and $V_k(s_k^{(CP)})$ as follows:

Step 1. If all inequalities are strictly satisfied, i.e., $\sum_{i \in I} w_i(x_k) = 0$, then $V_k(s_k^{(CP)}) = W_k$ and $s_k^{(CP)} = \alpha_k d_k^{(CP)}$, where $\alpha_k$ is given by (6.4).

Return.

Step 2. Compute the normalized negative gradient direction:

$$
d_k^{(CP)} = \frac{-(C'_k)^T W_k C_k}{\| (C'_k)^T W_k C_k \|}.
$$

Step 3. Solve

$$
\text{minimize } q^j_k(\alpha) = \frac{1}{2} \| V_{jk} (\alpha C'_k d_k^{(CP)} + C_k) \|^2
$$

subject to $|\alpha| \leq \Delta_k$

for the new iterate $\alpha_{j+1}$.

- Compute the Newton step

$$
\alpha_{j+1} = \frac{-C'_k V_{jk} C'_k d_k^{(CP)}}{(d_k^{(CP)})^T C'_k V_{jk} C'_k d_k^{(CP)}}.
$$

- Determine if the trust region is binding; if $\alpha_{j+1} > \Delta_k$, then $\alpha_{j+1} = \Delta_k$.

Step 4. Evaluate $V_k(\alpha_{j+1})$ as in (6.1).
Step 5. Check for convergence.
• If \( V_k(\alpha_{j+1}) = V_k(\alpha_j) \) or \( (\alpha_{j+1} = \Delta_k) \), then \( \alpha_{j+1} \) solves (6.5).
  Else
  — Compute the gradient of the new quadratic model,
  \[
  \nabla q^{j+1}(\alpha_{j+1}) = (C_k + \alpha_{j+1}C'_k d_k) V_k(\alpha_{j+1}) C'_k d_k.
  \]
  — If \( (\nabla q^{j+1}(\alpha_{j+1}) = 0) \), then \( \alpha_{j+1} \) solves (6.5).
  Else go to Step 3.

Step 6. If \( V_k(s^{\text{gp}}) \neq W_k \), then compute \( s^{\text{gp}} = -\alpha_k (C'_k)^T V_k(s^{\text{gp}}) C_k \).

6.3. Solution of the trust-region subproblem. We now consider an algorithm that approximates the solution to the following trust-region subproblem:

\[
\begin{aligned}
\text{minimize} & \quad \frac{1}{2} \| V_k(s^{\text{gp}})(C'_k s + C_k) \|^2 \\
\text{subject to} & \quad \| s \| \leq \Delta_k.
\end{aligned}
\]

**Algorithm 6.3.** Multimodel Dogleg Step.

Compute the generalized Cauchy step \( s^{\text{gp}} \) and \( V(s^{\text{gp}}) \):

If \( (\| s^{\text{gp}} \| = \Delta_k) \), then \( s_k = s^{\text{gp}} \). (* trust region was binding *)

Else if \( (\nabla q_k(s^{\text{gp}}) = 0) \), then \( s = s^{\text{gp}} \).

Else
  * Compute \( s^f \), the minimum norm solution to
    \[
    \text{minimize} \quad \frac{1}{2} \| V(s^{\text{gp}})(C_k s + C'_k) \|^2
    \]
  * If \( (\| s^f \| \leq \Delta_k) \), then \( s_k = s^f \)
  * Else dogleg between \( s^{\text{gp}} \) and \( s^f \)

6.4. Accepting the steps. Let \( s_k \) be a trial step computed by the algorithm. We test whether the point \( x_{k+1} = x_k + s_k \) is making progress toward the feasible region. We define the actual reduction in moving from \( x_k \) to \( x_{k+1} \) to be

\[
\text{Ared}_k = \Phi_k - \Phi_{k+1},
\]

\[
= \frac{1}{2} \left[ \| W_k C_k \|^2 - \| W_{k+1} C_{k+1} \|^2 \right].
\]

The predicted reduction will be

\[
\text{Pred}_k = \frac{1}{2} \left[ \| W_k C_k \|^2 - \| V_k(s^{\text{gp}})(C_k + C'_k s_k) \|^2 \right].
\]

From the way of computing the trial step, the predicted reduction is defined to produce a fraction of the Cauchy decrease in \( \Phi \) at \( x_k \), which means that \( \text{Pred}_k > 0 \). Hence, the step is accepted if \( \frac{\text{Ared}_k}{\text{Pred}_k} \geq \eta_1 \) where \( \eta_1 \in (0, 1) \).

Our rule for accepting the step and updating the trust-region radius for this algorithm is the same as in section 5.3.

7. Convergence results for the single-model algorithm. In this section, we will use the convergence theory for trust-region methods provided in Moré [23] to show that the Levenberg-Marquardt approach to the solution of (1.1) is globally convergent to a first-order stationary point under reasonable assumptions on \( C(x) \).
We make the following assumption on the sequence of iterates \( \{x_k\} \) generated by the single-model algorithm.

**Assumption 3.** For all \( k \), \( x_k \) and \( x_k + s_k \in \Omega \).

In order to apply Theorem 4.14 from Moré [23], we need to establish that \( \Phi(x) \) is bounded below, \( \nabla \Phi(x) \) is uniformly continuous, and the model Hessian is uniformly bounded. From (2.2) and (2.3), it is obvious that \( \Phi(x) \) is bounded below by 0. We obtain the following convergence result.

**Lemma 7.1.** Let Assumptions 1, 2, and 3 hold. Then,

\[
\lim_{k \to \infty} \| \nabla \Phi(x_k) \| = 0.
\]

**Proof.** A straightforward calculation from (5.1) yields

\[
\nabla q(x_k) = C'(x_k)W(x_k)C(x_k) = \nabla \Phi(x_k).
\]

Lemma 4.1 shows that \( \nabla \Phi \) is Lipschitz continuous, and thus, uniformly continuous.

Boundedness of the Hessian of the quadratic model can be established in the Frobenius norm in the following manner:

\[
\| \nabla^2 q(x_k) \|_F \leq \| C'_E^T C'_E \|_F + \| C'_I(x_k)W_I(x_k)C'_I(x_k) \|_F \\
\leq \| C'_E \|_F^2 + \| C'_I \|_F^2 \\
\leq \beta^2_E + \beta^2_I.
\]

The lemma then follows from Theorem 4.14 in Moré [23].

**8. Convergence results for the multimodel algorithm.** In this section we start by proving some intermediate lemmas needed for global convergence. Then we prove our main global convergence results.

We add to our list of assumptions the following assumption on the sequence of iterates \( \{x_k\} \) generated by the multimodel algorithm.

**Assumption 3'.** For all \( k \), \( x_k + s_{k}^{\text{gp}} \), \( x_k + s_{k}^{\text{gp}_{k}} \), and \( x_k + s_k \in \Omega \).

We start with the following lemma, which is needed in the proof of Lemma 8.2.

**Lemma 8.1.** Let Assumptions 1, 2, and 3' hold. Suppose that at any given iteration \( k \) there exists an \( i \in I \) such that \( |c_{ik}| > 0 \). If \( \Delta_k \) satisfies

\[
\Delta_k \leq \frac{1}{2\beta} \min_{c_{ik} \neq 0} |c_{ik}|,
\]

where \( \beta \) is as in (4.3), then

\[
W_{k+1} = W_k - B_k
\]

and

\[
V_k(s_k^{\text{gp}}) = W_k - \bar{B}_k,
\]

where \( B_k \) is a 0–1 diagonal matrix whose diagonal elements are

\[
b_i = \begin{cases} 
1 & \text{if } i \in I, \ c_{ik} = 0, \text{ and } c_{ik+1} < 0, \\
0 & \text{otherwise},
\end{cases}
\]
and $\tilde{B}_k$ is a 0–1 diagonal matrix whose diagonal elements are

$$\bar{b}_i = \begin{cases} 
1 & \text{if } i \in I, \ c_{ik} = 0, \text{ and } c_{ik}^t s_k^{\text{rep}} < 0, \\
0 & \text{otherwise.}
\end{cases}$$

Proof. Let $i$ index a constraint such that $c_{ik} \neq 0$. We will show that, if (8.1) holds, $c_{ik+1}$ has the same sign as $c_{ik}$.

From the hypothesis, $c_{ik+1} = c_{ik} + c_{ik}^t (x_k + \xi s_k) s_k$, where $\xi \in (0, 1)$. Therefore, if $c_{ik} > 0$, then $c_{ik+1} \geq c_{ik} - \|c_{ik}^t (x_k + \xi s_k)\| |s_k|$, and because $|s_k| \leq \Delta_k$ satisfies (8.1), we have $c_{ik+1} > \frac{c_{ik}}{2} > 0$. On the other hand, if $c_{ik} < 0$, then $c_{ik+1} \leq c_{ik} + \|c_{ik}^t (x_k + \xi s_k)\| |s_k|$, and again, because $|s_k|$ satisfies (8.1), we have $c_{ik+1} < \frac{-c_{ik}}{2} < 0$.

Therefore, for all the above cases, if $w_{ik} = 1$, then $w_{ik+1} = 1$ and hence $b_{ik} = 0$.

Now consider the case where $c_{ik} = 0$. In this case $w_{ik} = 1$. Therefore, if $c_{ik+1} \geq 0$, then $w_{ik+1} = 1$, and hence $b_{ik} = 0$. Thus, the only case where $b_{ik} = 1$ is when $c_{ik} = 0$ and $c_{ik+1} < 0$. Hence, (8.2) easily follows. Similarly, we can show that (8.3) holds.

This completes the proof. \( \quad \Box \)

The following lemma shows, for a fixed iterate $k$, how accurate our definition of predicted reduction is as an approximation to the actual reduction. This lemma is used in the proof of Theorem 8.5.

Lemma 8.2. Let Assumptions 1, 2, and 3' hold. For a fixed $k$ and varying $\Delta_k$, there exists a positive constant $a_1$, which depends on $k$, such that, if $\Delta_k$ satisfies (8.1), then

$$|A_{red_k} - Pred_k| \leq a_1 |s_k|^2.$$

Proof. If there is no index $i$ such that $i \in I$ and $c_{ik} > 0$ or $i \in E$ and $|c_{ik}| > 0$, then the point $x_k$ is a solution and the algorithm is terminated and there is nothing to prove.

Consider the case when there is at least one index $i \in I$ with $c_{ik} > 0$ or $i \in E$ with $|c_{ik}| > 0$. Because of the assumption that $\Delta_k$ satisfies (8.1) and using the above lemma, we have

$$A_{red_k} = \frac{1}{2} [C_k^T W_k C_k - C_{k+1} W_{k+1} C_{k+1}]$$

$$= \frac{1}{2} [C_k^T W_k C_k - C_{k+1}^T (W_k - B_k) C_{k+1}]$$

and

$$Pred_k = \frac{1}{2} [C_k^T W_k C_k - (C_k + C_k^t s_k)^T V_k (s_k^{rep})(C_k + C_k^t s_k)]$$

$$= \frac{1}{2} [C_k^T W_k C_k - (C_k + C_k^t s_k)^T (W_k - B_k)(C_k + C_k^t s_k)].$$

Thus,

$$|A_{red_k} - Pred_k| = \frac{1}{2} |C_{k+1}^T (W_k - B_k) C_{k+1} - (C_k + C_k^t s_k)^T (W_k - B_k)(C_k + C_k^t s_k)|.$$ But, because $B_k C_k = 0$ and $\tilde{B}_k C_k = 0$, we have

$$|A_{red_k} - Pred_k| = \frac{1}{2} |(C_k + C_k^t (x_k + \xi s_k) s_k)^T W_k (C_k + C_k^t (x_k + \xi s_k) s_k)$$

$$- s_k^T C_k^t (x_k + \xi s_k)^T B_k C_k^t (x_k + \xi s_k) s_k$$

$$- (C_k + C_k^t s_k)^T W_k (C_k + C_k^t s_k) + s_k^T C_k^t \tilde{B}_k C_k^t s_k|.$$
Using Assumptions 1, 2, and 3’ we can easily obtain

\[ |Ared_k - Pred_k| \leq a_1 ||s_k||^2. \]

This completes the proof. \( \square \)

In the above lemma we bound the difference between \( Ared_k \) and \( Pred_k \) by a constant \( a_1 \) times \( ||s_k||^2 \). This constant depends on \( k \). This lemma can be used for a fixed iterate \( k \), but it cannot be used, with the same constant \( a_1 \), across all iterates.

The following lemma gives a bound to \( |Ared_k - Pred_k| \) that can be used across all iterates. It also shows how accurate our definition of predicted reduction is as an approximation to the actual reduction. It says that \( |Ared_k - Pred_k| = o(\Delta_k) \). This lemma is used in the proof of Theorem 8.6.

**Lemma 8.3.** Let Assumptions 1, 2, and 3’ hold. Then, as \( \Delta_k \to 0 \),

\[ |Ared_k - Pred_k| \leq o(\Delta_k). \]  

**Proof.** From the definition of \( Ared_k \) and \( Pred_k \), we have

\[ |Ared_k - Pred_k| = \frac{1}{2} \left| \|W_k C_k\|^2 - \|W_{k+1} C_{k+1}\|^2 - \|W_k C_k\|^2 \right. \]

\[ + \left. ||V_k(s_k^{\text{gcp}})(C_k + C_k's_k)||^2 \right| \]

\[ \leq \left| \|W_k C_k\|^2 - \|W_{k+1} C_{k+1}\|^2 - \|W_k C_k\|^2 \right. \]

\[ + \left. ||W_k(C_k + C_k's_k)||^2 \right| \]

\[ + \left| ||W_k(C_k + C_k's_k)||^2 - ||V_k(s_k^{\text{gcp}})(C_k + C_k's_k)||^2 \right| \]

\[ + \left| ||V_k(s_k^{\text{gcp}})(C_k + C_k's_k)||^2 - ||V_k(s_k^{\text{gcp}})(C_k + C_k's_k)||^2 \right| \]

\[ = I + II + III, \]

where

\[ I = \left| \|W_{k+1} C_{k+1}\|^2 - \|W_k C_k\|^2 + \|W_k C_k\|^2 - \|W_k(C_k + C_k's_k)||^2 \right|, \]

\[ II = \left| ||W_k(C_k + C_k's_k)||^2 - ||V_k(s_k^{\text{gcp}})(C_k + C_k's_k)||^2 \right|, \] and

\[ III = \left| ||V_k(s_k^{\text{gcp}})(C_k + C_k's_k)||^2 - ||V_k(s_k^{\text{gcp}})(C_k + C_k's_k)||^2 \right|. \]

From the continuity, we can easily show that

\[ I \leq O(||s_k||^2) = O(\Delta_k^2) = o(\Delta_k). \]

On the other hand,

\[ II = \left| \|W_k(C_k + C_k's_k)||^2 - ||V_k(s_k^{\text{gcp}})(C_k + C_k's_k)||^2 \right|, \]

\[ = \left| \|W_k(C_k + C_k's_k)||^2 - 2\Psi_k(0) + 2\Psi_k(0) - 2\Psi_k(s_k^{\text{gcp}}) \right|, \]

where \( \Psi_k \) is given by (6.2). Using the mean-value theorem on the real function \( \Psi_k(t s_k^{\text{gcp}}) \) on \([0,1]\), we have

\[ II \leq \left| \|W_k(C_k + C_k's_k)||^2 - \|W_k C_k||^2 + 2\nabla \Psi_k(s_k^{\text{gcp}})^T s_k^{\text{gcp}} \right| \]

\[ = \left| 2(C_k + C_k's_k)^T W_k C_k's_k + 2\nabla \Psi_k(s_k^{\text{gcp}})^T s_k^{\text{gcp}} \right| \]

\[ \leq 2 \left| \nabla \Psi_k(0)^T s_k - \nabla \Psi_k(s_k^{\text{gcp}})^T s_k^{\text{gcp}} \right| + O(||s_k||^2) \]

\[ \leq 2 \left| \nabla \Psi_k(0)^T s_k - \nabla \Psi_k(0)^T s_k^{\text{gcp}} + \nabla \Psi_k(0)^T s_k^{\text{gcp}} - \nabla \Psi_k(s_k^{\text{gcp}})^T s_k^{\text{gcp}} \right| \]

\[ + O(||s_k||^2) \]

\[ \leq 2 \left| \nabla \Psi_k(0)^T s_k - s_k^{\text{gcp}} \right| + O(||s_k||^2 + ||s_k^{\text{gcp}}||^2) \]

\[ \leq 2 ||\nabla \Psi_k(0)|| \left| ||s_k - s_k^{\text{gcp}}|| + O(||s_k||^2 + ||s_k^{\text{gcp}}||^2) \right|, \]
where $\xi, \bar{\xi} \in (0, 1)$. Also,

\begin{equation}
\|s_k - s_k^{\text{gcp}}\| = \left\| \frac{s_k}{\|s_k\|} - \frac{s_k^{\text{gcp}}}{\|s_k\|} \right\| s_k = \|u_k - \nu_k \bar{u}_k\| s_k,
\end{equation}

where $u_k$ is a unit vector in the direction of the vector $s_k$, $\bar{u}_k$ is a unit vector in the direction of the vector $s_k^{\text{gcp}}$, and $\nu_k = \|s_k^{\text{gcp}}\| / \|s_k\|$. We have $\|\nabla \Psi_k(0)\| = \|C_k' W_k C_k\|$. Let $\Delta_k \to 0$, $s_k \to 0$. First consider the case $\|C_k' W_k C_k\| \to 0$. The definition of the dogleg step implies that $\|s_k^{\text{gcp}}\| \leq \|s_k\|$ (see also Dennis and Schnabel [13], Sec. 6.4.2). Hence, $\nu_k \leq 1$ and

\begin{equation}
\|C_k' W_k C_k\| \|u_k - \nu_k \bar{u}_k\| \leq 2 \|C_k' W_k C_k\| \to 0.
\end{equation}

On the other hand, if $\|C_k' W_k C_k\|$ is bounded away from zero, then $\|s_k^{\text{gcp}}\| = \Delta_k$ for all sufficiently small $\Delta_k$ (Algorithm 6.2). Since a dogleg strategy is employed to compute $s_k$, this implies $s_k = s_k^{\text{gcp}}$ for all sufficiently small $\Delta_k$ (Algorithm 6.3). Thus, $\|u_k - \nu_k \bar{u}_k\| = 0$ for all sufficiently small $\Delta_k$. Therefore, in either case,

\begin{equation}
\|\nabla \Psi_k(0)\| \|s_k - s_k^{\text{gcp}}\| = \|C_k' W_k C_k\| \|u_k - \nu_k \bar{u}_k\| \|s_k\| = o(\|s_k\|).
\end{equation}

Hence, we have $\|C_k' W_k C_k\| \|s_k - s_k^{\text{gcp}}\| \leq \hat{\epsilon}_k \|s_k\|$, where $\hat{\epsilon}_k \to 0$ as $\Delta_k \to 0$. Therefore, we can write

\begin{equation}
\Pi \leq \hat{\epsilon}_k \|s_k\| + O(\|s_k^{\text{gcp}}\|^2) + O(\|s_k\|^2) = o(\Delta_k).
\end{equation}

Finally,

\begin{align*}
\Pi & = \left\| V_k (s_k^{\text{gcp}}(C_k + C_k' s_k^{\text{gcp}})) - V_k (s_k^{\text{gcp}})(C_k + C_k' s_k^{\text{gcp}})^2 \right\| \\
& = \left\| C_k' V_k (s_k^{\text{gcp}}) C_k + 2C_k' V_k (s_k^{\text{gcp}}) C_k' s_k^{\text{gcp}} + (s_k^{\text{gcp}})^T C_k' V_k (s_k^{\text{gcp}}) C_k' s_k^{\text{gcp}}
- C_k' V_k (s_k^{\text{gcp}}) C_k - 2C_k' V_k (s_k^{\text{gcp}}) C_k' s_k^{\text{gcp}} - C_k' V_k (s_k^{\text{gcp}}) C_k' s_k^{\text{gcp}} \right\| \\
& \leq 2 \|C_k' V_k (s_k^{\text{gcp}}) C_k\| \|s_k - s_k^{\text{gcp}}\| + O(\|s_k^{\text{gcp}}\|^2) + O(\|s_k\|^2).
\end{align*}

Now using an argument similar to the one we used in (8.8), we can also write

\begin{equation}
\|C_k' V_k (s_k^{\text{gcp}}) C_k\| \|s_k - s_k^{\text{gcp}}\| \leq \hat{\epsilon}_k \|s_k\|,
\end{equation}

where $\hat{\epsilon}_k \to 0$ as $\Delta_k \to 0$. So,

\begin{equation}
\Pi \leq \hat{\epsilon}_k \|s_k\| + O(\Delta_k^2) = o(\Delta_k).
\end{equation}

Combining (8.6), (8.9), and (8.10), we obtain (8.5). This completes the proof of the lemma.

**Lemma 8.4.** At any iteration $k$, we have

\begin{equation}
Pred_k \geq \|C_k' W_k C_k\| \min \{\Delta_k, a_2 \|C_k' W_k C_k\| \}
\end{equation}

where $a_2$ is a constant that does not depend on $k$. 

---

"TWO ALGORITHMS FOR SYSTEMS OF EQUALITIES AND INEQUALITIES" 307
Proof. From the definition of \( \text{Pred}_k \) and \( V_k(s_{\text{gcp}}^k) \), we have
\[
\text{Pred}_k = \frac{1}{2} \left[ \|W_k C_k\|^2 - \|V_k(s_{\text{gcp}}^k)(C_k + C_k' s_k)\|^2 \right]
\geq \frac{1}{2} \left[ \|W_k C_k\|^2 - \|V_k(s_{\text{gcp}}^k)(C_k + C_k' s_k)\|^2 \right]
\geq \frac{1}{2} \left[ \|W_k C_k\|^2 - \|W_k(C_k + C_k' s_k)\|^2 \right].
\]

The rest of the proof is straightforward. See, for example, Powell [29].

The following theorem shows that the algorithm is well defined in the sense that it will never loop ad infinitum without finding an acceptable step.

Theorem 8.5. Let Assumptions 1, 2, and 3 hold. At any iteration \( k \), either the point \( x_k \) satisfies \( \nabla \Phi(x_k) = 0 \) or an acceptable step will be found.

Proof. In the proof of this lemma we use the notation \( k_j \) to mean the \( j \)th unacceptable trial step of iteration \( k \).

If the point \( x_k \) satisfies \( \|C_k' W_k C_k\| = 0 \), then it is a stationary point to the problem and there is nothing to prove.

Assume that \( \nabla \Phi(x_k) \equiv \|C_k' W_k C_k\| \neq 0 \) and suppose that the algorithm loops infinitely without finding an acceptable step. Hence all the trial steps are rejected and we obtain, for all \( j \),
\[
(1 - \eta_j) < \left| \frac{\text{Ared}_{k_j}}{\text{Pred}_{k_j}} - 1 \right|.
\]

For sufficiently large \( j \), and because \( \|C_k' W_k C_k\| \neq 0 \), the trust region will have been reduced sufficiently so that inequality (8.11) will have the form
\[
\text{Pred}_{k_j} \geq \|C_k' W_k C_k\| \Delta_{k_j}
\]
and \( \Delta_{k_j} \) will satisfy (8.1). Hence, Lemma 8.2 implies
\[
|\text{Ared}_k - \text{Pred}_k| \leq a_1 \|s_k\|^2.
\]

The above two inequalities imply that for \( j \) sufficiently large, we have
\[
\frac{|\text{Ared}_{k_j} - \text{Pred}_{k_j}|}{\text{Pred}_{k_j}} = \frac{|\text{Ared}_{k_j} - \text{Pred}_{k_j}|}{\text{Pred}_{k_j}} \leq O(\Delta_{k_j}).
\]

This means that, as \( j \to \infty \), \( \Delta_{k_j} \to 0 \) and \( \frac{|\text{Ared}_{k_j} - \text{Pred}_{k_j}|}{\text{Pred}_{k_j}} \to 0 \). This contradicts (8.12). Hence, \( j \) cannot go to infinity. But this contradicts the supposition that the algorithm loops infinitely without finding an acceptable step and means that, after finitely many rejected trial steps, an acceptable one will be found. This completes the proof. \( \Box \)

Now we present our main global convergence result. We show that the algorithm will converge to a stationary point of problem (1.2). In particular, we show that \( \lim_{k \to \infty} \nabla \Phi(x_k) = 0 \), where \( \Phi(x) \) is given by (2.4). This is equivalent to proving that \( \lim_{k \to \infty} \|C_k' W_k C_k\| = 0 \).

Theorem 8.6. Let Assumptions 1, 2, and 3 hold. The algorithm generates a sequence of points \( \{x_k\} \) that satisfies
\[
\lim_{k \to \infty} \nabla \Phi(x_k) = 0.
\]
Proof. Suppose that \( \limsup_{k \to \infty} \| C'_{k}^T W_{k} C_{k} \| = \epsilon_0 > 0 \). Then there exists an infinite sequence of indices \( \{ k_j \} \), such that \( \| C'_{k_j}^T W_{k_j} C_{k_j} \| > \frac{\epsilon_0}{2} \) for all \( k \in \{ k_j \} \).

Let \( \hat{k} \) be such that \( \hat{k} \in \{ k_j \} \). Since from Lemma 4.1, \( \nabla \Phi \) is Lipschitz continuous in \( \Omega \), we have, for any \( x \in \Omega \),

\[
\| C'(x)^T W(x) C(x) \| \geq \| C'_{\hat{k}}^T W_{\hat{k}} C_{\hat{k}} \| - a_0 \| x - x_{\hat{k}} \|.
\]

This implies that for all \( x \) that satisfy \( \| x - x_{\hat{k}} \| \leq \frac{\| C'_{\hat{k}}^T W_{\hat{k}} C_{\hat{k}} \|}{2a_0} = \sigma \), we have

\[
\| C'(x)^T W(x) C(x) \| \geq \frac{1}{2} \| C'_{\hat{k}}^T W_{\hat{k}} C_{\hat{k}} \| > \frac{\epsilon_0}{4}.
\]

Consider the ball \( B_\sigma = \{ x : \| x - x_{\hat{k}} \| \leq \sigma \} \).

If \( x_k \in B_\sigma \) for all \( k \geq \hat{k} \), then from (8.11), we have

\[
(8.13) \quad \text{Pred}_k \geq \| C'_{k}^T W_{k} C_{k} \| \min \{ \Delta_k, a_2 \| C'_{k}^T W_{k} C_{k} \| \}.
\]

Because \( x_k \in B_\sigma \), we have \( \| C'_{k}^T W_{k} C_{k} \| \geq \frac{\epsilon_0}{4} \). Hence, for all \( k \geq \hat{k} \)

\[
(8.14) \quad \text{Pred}_k \geq \frac{\epsilon_0}{4} \min \{ \Delta_k, a_2 \frac{\epsilon_0}{4} \}.
\]

If there were no acceptable steps for all \( k \geq \hat{k} \), a contradiction to Theorem 8.5 would arise. Hence there exists an infinite sequence of acceptable steps. For any such \( k \),

\[
\Phi_k - \Phi_{k+1} = A\text{red}_k \geq \eta_1 \text{Pred}_k.
\]

Since \( \Phi_k \) is bounded below and decreasing, \( \Phi_k - \Phi_{k+1} \to 0 \), and we obtain, using the above inequality and (8.14),

\[
(8.15) \quad \liminf_{k \to \infty} \Delta_k = 0.
\]

Hence, using Lemmas 8.3 and 8.4, the above limit implies that

\[
\lim_{k \to \infty} \left| \frac{A\text{red}_k}{\text{Pred}_k} - 1 \right| = \lim_{\Delta_k \to 0} \left| \frac{A\text{red}_k - \text{Pred}_k}{\text{Pred}_k} \right| = 0.
\]

However, the updating rules for \( \Delta_k \) increase \( \Delta_k \) if \( \frac{A\text{red}_k}{\text{Pred}_k} > \eta_2 \). Thus \( \Delta_k \) cannot converge to zero. But this contradicts (8.15) and means that all the iterates cannot stay inside \( B_\sigma \).

Let \( l + 1 \) be the first index greater than \( \hat{k} \) such that \( x_{l+1} \) does not lie inside the ball \( B_\sigma \). Hence

\[
\Phi_{\hat{k}} - \Phi_{l+1} = \sum_{k=\hat{k}}^{l} \{ \Phi_k - \Phi_{k+1} \} = \sum_{k=\hat{k}}^{l} A\text{red}_k
\]

\[
\geq \eta_1 \sum_{k=\hat{k}}^{l} \text{Pred}_k \geq \eta_1 \frac{\epsilon_0}{4} \min \left[ \sum_{k=\hat{k}}^{l} \Delta_k, a_2 \frac{\epsilon_0}{4} \right]
\]

\[
\geq \eta_1 \frac{\epsilon_0}{4} \min \left[ \frac{\epsilon_0}{4a_0}, a_2 \frac{\epsilon_0}{4} \right].
\]

As \( l \) and \( \hat{k} \) go to infinity, \( \Phi_{\hat{k}} - \Phi_{l+1} \) goes to zero, which contradicts the supposition that \( \epsilon_0 > 0 \). This proves the theorem. \( \square \)
9. Numerical examples. In this section, we report our preliminary numerical experience with the two algorithms. The numerical experiments were done on a Sun 4/490 workstation running SunOS operating system release 4.1.3 with 64 megabytes of memory. The programs were written in MATLAB and run under MATLAB version 4.2a with machine epsilon about $10^{-16}$. More numerical investigation with larger dimensional problems is needed to fully understand the behavior of the two algorithms, but the results given here are very encouraging when compared to LANCELOT on the same problems.

9.1. Algorithm parameters. Successful termination means that the termination condition of the algorithm, $\|C_k^TW_kC_k\| \leq \varepsilon_{tol} = 10^{-6}$, is met. On the other hand, if the algorithm generates a step of length less than $10^{-10}$, the number of iterations is greater than 75, or the number of function evaluations is greater than 100, then it is considered an unsuccessful termination.

The initial trust-region radius for the two algorithms is set to $\Delta_0 = \|s_0^P\|$. The values of the parameters used for updating the trust-region radius in sections 5.3 and 6.4 are $\eta_1 = 10^{-4}$, $\eta_2 = 0.1$, $\eta_3 = 0.25$, $\eta_4 = 0.75$, $\alpha_1 = 0.3$, $\alpha_2 = 2$, and $\alpha_3 = 4$.

9.2. Numerical results. In this section, we report the numerical results for our two trust-region algorithms described in sections 5 and 6. The problems that we tested are the constraint sets of a subset of the Constrained and Unconstrained Testing Environment (the CUTE collection [3]). Some of these problems are from Schittkowski [33]. See also Hock and Schittkowski [18] and Schittkowski [33] for more test problems. We used a MATLAB interface written by Branch (see [4]).

The symbol * is added to the name of some problems to indicate that the corresponding problems have been modified by adding infeasible simple bounds on their variables. The results are summarized in Tables 9.1 and 9.2. Note that $m > n$ for several problems.

In Table 9.1, columns 1–4 give the data of the problem. In particular, the first column gives the problem name. The second column gives the dimension (number of variables) of the problem. The third and fourth columns give the number of equalities and the number of inequalities, respectively. In the fifth and sixth columns we list, respectively, the average number of iterations and the average number of function evaluations needed by the single-model algorithm to converge from different starting points to points that satisfy the stopping criterion. In the seventh and eighth columns we list the corresponding results for the multimodel algorithm. The average number of starting points tried for each test problem was about seven points, and they were chosen randomly.

We also compared our two trust-region algorithms against LANCELOT (release A). LANCELOT is a Fortran package for large-scale nonlinear optimization written by Conn, Gould, and Toint [11].

The test problems that we used are from the CUTE collection with the default starting points. To make the comparison simpler, we selected the subset of the CUTE test problems that have no objective function. LANCELOT in this case will find a feasible point. We note here that we used LANCELOT with all its parameters set to their default values.

In Table 9.2, we report the results of the single-model algorithm and the results obtained using LANCELOT. For the test problems used in Table 9.2, the results obtained by the multimodel algorithm are either the same as those obtained by the single-model algorithm or there is a slight edge in favor of the multimodel algorithm, so the results would even be slightly more in our favor had we given the multimodel
results. Remember that our single and multimodel algorithms are identical when there are no inequalities.

In the second table, we report the number of function evaluations and the number of gradient evaluations used. Indeed, the numbers of function and gradient evaluations are good measures for the number of trial steps and the number of acceptable steps. In particular, the number of function evaluations is greater by one than the number of trial steps computed by the algorithm. The number of gradient evaluations is greater by one than the number of acceptable steps used by the algorithm to converge from the default starting points to points that satisfy the stopping criterion.

Again, in Table 9.2, columns 1–4 give the data of the problem. In columns 5 and 6, we list, respectively, the number of function evaluations and the number of gradient evaluations for the single-model algorithm. In columns 7 and 8, we list the corresponding results for the LANCELOT. Note that the maximum number of function evaluations allowed by LANCELOT is 1000.

In most of the test problems reported in Table 9.2, the number of function evaluations and the number of gradient evaluations obtained by our trust-region algorithm are better than those obtained by LANCELOT. This gives some indication about the viability of our approach. However, we believe that our two algorithms need to be refined with efficiency in mind to be suitable for large-scale problems. We also expect that the roughly 10% better performance of the multimodel over the single-model algorithm would be larger in the large-scale case with many inequality constraints (see section 10).

10. Concluding remarks. We have introduced two new trust-region algorithms for finding a feasible point of a set of equalities and inequalities. A one-sided least-squares formulation of the problem is described. The formulation is free of arbitrary parameters and possesses sufficient smoothness to exploit the robustness of the two algorithms. The first algorithm is a single-model algorithm. The second one is a multimodel algorithm. Global convergence results for the two algorithms are presented. It is shown that these two algorithms are globally convergent to a first-order stationary point. Another novelty is that we have given a global convergence analysis for an algorithm based on a local model that does not match function or gradient information.

We point out that our global results give us only first-order stationary point convergence. Since we are using a least-squares formulation of problem (1.1), (i.e., solving problem (1.2)), there is the possibility that the algorithm will converge to a stationary point \(x^*\) with \(\Phi(x^*) > 0\). This can happen when the rank of the matrix \(W^*C'(x^*)\) is less than the number of equalities and the active inequalities at the solution \(x^*\).

We have reported preliminary numerical results with the two algorithms. For large-scale problems, computing an accurate minimum norm solution in high dimensions is probably too costly. Nevertheless, adapting an iterative method with a truncation procedure can reduce the cost of the minimum norm solution. We believe that there is considerable scope for modifying and adapting the basic ideas presented in this paper to the large-scale setting. A more comprehensive computational investigation of the two algorithms, particularly for large problems, needs to be done. It will be presented in a subsequent paper.

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

The algorithms that were developed in this paper are for finding a feasible point of a set of equalities and inequalities. An important question, which is also a research
Table 9.1

Computational results for the two algorithms.

<table>
<thead>
<tr>
<th>Prob. name</th>
<th>n</th>
<th>E</th>
<th>I</th>
<th>iter.</th>
<th>nfunc</th>
<th>iter.</th>
<th>nfunc</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGG</td>
<td>163</td>
<td>36</td>
<td>615</td>
<td>42</td>
<td>61</td>
<td>51</td>
<td>76</td>
</tr>
<tr>
<td>BDVALUE</td>
<td>12</td>
<td>10</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>DALLAS</td>
<td>196</td>
<td>151</td>
<td>392</td>
<td>8.5</td>
<td>11.3</td>
<td>7</td>
<td>8.8</td>
</tr>
<tr>
<td>EIGEN</td>
<td>110</td>
<td>110</td>
<td>110</td>
<td>18</td>
<td>24</td>
<td>17</td>
<td>21.5</td>
</tr>
<tr>
<td>HATFLDG</td>
<td>25</td>
<td>25</td>
<td>40</td>
<td>7</td>
<td>8</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>HS6*</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>18</td>
<td>37.6</td>
<td>18</td>
<td>37.6</td>
</tr>
<tr>
<td>HS7*</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>16.6</td>
<td>17.6</td>
<td>16.6</td>
<td>17.6</td>
</tr>
<tr>
<td>HS10</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>11.2</td>
<td>12.2</td>
<td>11.2</td>
<td>12.2</td>
</tr>
<tr>
<td>HS11</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>9.8</td>
<td>12.2</td>
<td>9.8</td>
<td>12.2</td>
</tr>
<tr>
<td>HS12</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>7.6</td>
<td>8.6</td>
<td>7.6</td>
<td>8.6</td>
</tr>
<tr>
<td>HS14</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>9.6</td>
<td>11.2</td>
<td>11</td>
<td>11.4</td>
</tr>
<tr>
<td>HS22</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>10.6</td>
<td>13</td>
<td>9.6</td>
<td>12</td>
</tr>
<tr>
<td>HS29</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>7.2</td>
<td>8.2</td>
<td>7.2</td>
<td>8.2</td>
</tr>
<tr>
<td>HS40*</td>
<td>4</td>
<td>3</td>
<td>8</td>
<td>16.8</td>
<td>19.6</td>
<td>16.6</td>
<td>20</td>
</tr>
<tr>
<td>HS43</td>
<td>4</td>
<td>0</td>
<td>3</td>
<td>10.8</td>
<td>14</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>HS60</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>22.2</td>
<td>26.4</td>
<td>14.4</td>
<td>17.6</td>
</tr>
<tr>
<td>HS78*</td>
<td>5</td>
<td>2</td>
<td>10</td>
<td>16</td>
<td>22.8</td>
<td>15.2</td>
<td>21.4</td>
</tr>
<tr>
<td>HS80</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>14</td>
<td>17.2</td>
<td>15.4</td>
<td>19</td>
</tr>
<tr>
<td>HS99EXP</td>
<td>31</td>
<td>21</td>
<td>29</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>18</td>
</tr>
<tr>
<td>HS113</td>
<td>10</td>
<td>0</td>
<td>8</td>
<td>12.4</td>
<td>14.8</td>
<td>19.8</td>
<td></td>
</tr>
<tr>
<td>HYDCAR20</td>
<td>99</td>
<td>99</td>
<td>0</td>
<td>17</td>
<td>22</td>
<td>17</td>
<td>22</td>
</tr>
<tr>
<td>HYDCAR6</td>
<td>29</td>
<td>29</td>
<td>0</td>
<td>8</td>
<td>9</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>LEAKNET</td>
<td>156</td>
<td>153</td>
<td>82</td>
<td>7.7</td>
<td>10</td>
<td>7.7</td>
<td>10</td>
</tr>
<tr>
<td>LINSFANH</td>
<td>97</td>
<td>33</td>
<td>194</td>
<td>34.3</td>
<td>46.7</td>
<td>38.3</td>
<td>54</td>
</tr>
<tr>
<td>NET1</td>
<td>48</td>
<td>38</td>
<td>75</td>
<td>14</td>
<td>17.5</td>
<td>14.5</td>
<td>18</td>
</tr>
<tr>
<td>PRODPL0</td>
<td>60</td>
<td>20</td>
<td>69</td>
<td>23.5</td>
<td>37.5</td>
<td>20</td>
<td>19.5</td>
</tr>
<tr>
<td>PRODPL1</td>
<td>60</td>
<td>20</td>
<td>69</td>
<td>34</td>
<td>56.5</td>
<td>39</td>
<td>49.5</td>
</tr>
<tr>
<td>QPNBLEND</td>
<td>83</td>
<td>43</td>
<td>114</td>
<td>36.5</td>
<td>57.5</td>
<td>17.5</td>
<td>23.5</td>
</tr>
<tr>
<td>HS 226</td>
<td>2</td>
<td>0</td>
<td>4</td>
<td>8.6</td>
<td>13</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>S227</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>11.4</td>
<td>13.2</td>
<td>8.8</td>
<td>10.4</td>
</tr>
<tr>
<td>S262</td>
<td>4</td>
<td>1</td>
<td>7</td>
<td>4</td>
<td>6</td>
<td>3.4</td>
<td>4.6</td>
</tr>
<tr>
<td>S263</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>12.6</td>
<td>14.6</td>
<td>11.4</td>
<td>13</td>
</tr>
<tr>
<td>S353</td>
<td>4</td>
<td>1</td>
<td>6</td>
<td>8</td>
<td>14.4</td>
<td>11.4</td>
<td>21.4</td>
</tr>
<tr>
<td>S354</td>
<td>4</td>
<td>0</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>SMBANK</td>
<td>117</td>
<td>64</td>
<td>234</td>
<td>12</td>
<td>17</td>
<td>8</td>
<td>11.5</td>
</tr>
<tr>
<td>SPANHYD</td>
<td>97</td>
<td>33</td>
<td>194</td>
<td>34</td>
<td>42.5</td>
<td>39</td>
<td>54.5</td>
</tr>
<tr>
<td>SSEBLIN</td>
<td>194</td>
<td>48</td>
<td>388</td>
<td>7</td>
<td>11</td>
<td>11</td>
<td>16</td>
</tr>
<tr>
<td>SSEBNLN</td>
<td>194</td>
<td>72</td>
<td>388</td>
<td>42</td>
<td>61</td>
<td>32</td>
<td>48</td>
</tr>
<tr>
<td>Totals</td>
<td></td>
<td></td>
<td></td>
<td>665.9</td>
<td>915</td>
<td>619.8</td>
<td>822.5</td>
</tr>
<tr>
<td>Averages</td>
<td></td>
<td></td>
<td></td>
<td>15.5</td>
<td>21.3</td>
<td>14.4</td>
<td>19.1</td>
</tr>
</tbody>
</table>

A related important question is how to generalize the multimodel theory developed in this paper to general nonlinear programming trust-region algorithms.

Acknowledgments. We wish to thank Amr El-Bakry for many useful discussions and comments and Luis Vicente for his careful reading of an earlier version.
of this paper. We also wish to thank three referees and the editor for their helpful reports.

REFERENCES


