SIAM J. OPTIM.
Vol. 14, No. 2, pp. 439–463
© 2003 Society for Industrial and Applied Mathematics

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

SOLVING KARUSH–KUHN–TUCKER SYSTEMS VIA THE TRUST REGION AND THE CONJUGATE GRADIENT METHODS[*]

HOUDUO QI[†], LIQUN QI[‡], AND DEFENG SUN[§]

Abstract. A popular approach to solving the Karush–Kuhn–Tucker (KKT) system, mainly arising from the variational inequality problem, is to reformulate it as a constrained minimization problem with simple bounds. In this paper, we propose a trust region method for solving the reformulation problem with the trust region subproblems being solved by the truncated conjugate gradient (CG) method, which is cost effective. Other advantages of the proposed method over existing ones include the fact that a good approximated solution to the trust region subproblem can be found by the truncated CG method and is judged in a simple way; also, the working matrix in each iteration is H, instead of the condensed H^TH, where H is a matrix element of the generalized Jacobian of the function used in the reformulation. As a matter of fact, the matrix used is of reduced dimension. We pay extra attention to ensure the success of the truncated CG method as well as the feasibility of the iterates with respect to the simple constraints. Another feature of the proposed method is that we allow the merit function value to be increased at some iterations to speed up the convergence. Global and superlinear/quadratic convergence is shown under standard assumptions. Numerical results are reported on a subset of problems from the MCPLIB collection [S. P. Dirkse and M. C. Ferris, *Optim. Methods Softw.*, 5 (1995), pp. 319–345].

Key words. variational inequality problem, constrained optimization, semismooth equation, trust region method, truncated conjugate gradient method, global and superlinear convergence

AMS subject classifications. 65H10, 90C30, 90C33

DOI. 10.1137/S105262340038256X

1. Introduction.

Given a continuously differentiable function $F : \mathbb{R}^n \to \mathbb{R}^n$ and twice continuously differentiable functions $h : \mathbb{R}^n \to \mathbb{R}^p$ and $g : \mathbb{R}^n \to \mathbb{R}^m$, we consider the following Karush–Kuhn–Tucker (KKT) system in (x, y, z):

$$\left.\begin{aligned} L(x,y,z) &= 0 \\ h(x) &= 0 \\ g(x) \geq 0, z \geq 0, z^T g(x) &= 0 \end{aligned}\right\}, \tag{1}$$

where L is called the Lagrangian of the functions F, g, and h and is defined by

$$L(x,y,z) := F(x) + \nabla h(x)y - \nabla g(x)z.$$

Due to its close relationship with the variational inequality problem (VIP) and the nonlinear constrained optimization problem (NLP) (in both cases, the functions h and g define the corresponding equality and inequality constraints, respectively), there is a growing interest in constructing efficient algorithms for (1); for the latest references, see [30, 11, 19]. In particular, Qi and Jiang [30] reformulate (1) to various

[*]Received by the editors December 15, 2000; accepted for publication (in revised form) April 30, 2003; published electronically October 14, 2003.
http://www.siam.org/journals/siopt/14-2/38256.html

[†]School of Mathematics, The University of New South Wales, Sydney 2052, Australia (hdqi@maths.unsw.edu.au). The research of this author was supported by the Australian Research Council.

[‡]Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong (maqilq@polyu.edu.hk). The research of this author was supported by the Hong Kong Research Grant Council and the Australian Research Council.

[§]Department of Mathematics, National University of Singapore, 2 Science Drive 2, Singapore 117543, Republic of Singapore (matsundf@nus.edu.sg). The research of this author was supported by grant R146-000-035-101 of the National University of Singapore.

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

semismooth equations, and local semismooth Newton methods are studied for these semismooth equations. One of these semismooth equations is based on the Fischer–Burmeister function [12]: $\varphi : \mathbb{R}^2 \to \mathbb{R}$ defined by $\varphi(a, b) := (a + b) - \sqrt{a^2 + b^2}$. An interesting property of φ_α is that $\varphi(a, b) = 0$ if and only if $a, b \geq 0, ab = 0$. Define

$$\phi(g(x), z) := (\varphi(g_1(x), z_1), \ldots, \varphi(g_m(x), z_m))^T \in \mathbb{R}^m,$$

and let $\Phi : \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m \to \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m$ be the equation operator

$$\Phi(w) := \Phi(x, y, z) := \begin{pmatrix} L(x, y, z) \\ h(x) \\ \phi(g(x), z) \end{pmatrix}.$$

Then $w^* = (x^*, y^*, z^*) \in \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m$ is a solution of (1) if and only if it solves the system of nonlinear equations $\Phi(w) = 0$. In other words, solving (1) is equivalent to finding a global solution of the problem

$$\min \ \Psi(w), \tag{2}$$

where

$$\Psi(w) := \frac{1}{2}\Phi(w)^T\Phi(w) = \frac{1}{2}\|\Phi(w)\|^2$$

denotes the natural merit function of the equation operator Φ. This unconstrained optimization approach has been used in [8, 9, 30] to develop some Newton-type methods for the solution of (1). Despite their strong theoretical and numerical properties, these methods may fail to find the unique solution of (1) arising from strongly monotone variational inequalities because the variable z is not forced to be nonnegative in [8, 9, 30]. For such an example, see [26, 11]. This, together with the fact that the variable z has to be nonnegative at a solution of (1), motivates Facchinei et al. [11] to investigate a quadratic programming (QP) based method for the solution of the constrained minimization problem

$$\min \ \Psi(w) \quad \text{subject to (s.t.)} \quad z \geq 0. \tag{3}$$

The subproblem in $d = (d_x, d_y, d_z) \in \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m$ solved at the current iteration $w^k = (x^k, y^k, z^k)$ (given $z^k \geq 0$) is of the type

$$\begin{array}{ll} \min & \nabla\Psi(w^k)^T d + \frac{1}{2} d^T(H_k^T H_k + \rho_k I)d \\ \text{s.t.} & z^k + d_z \geq 0, \end{array} \tag{4}$$

where $\rho_k > 0$, $H_k \in \partial\Phi(w^k)$, and $\partial\Phi(w^k)$ is the set of the generalized Jacobian of Φ at w^k in the sense of Clarke [4]. An inexact version of this QP-based method was provided by Kanzow [17] for the nonlinear complementarity problem (NCP) with an inexact solution of the QP subproblem being calculated by an interior-point method and the inexactness being measured in a similar way as described by Gabriel and Pang [14]. We emphasize that it is the interior-point method that guarantees the constraints to be nonviolated. Using an active-set strategy, Kanzow and Qi [19] proposed a QP-free method, which requires solving one system of linear equations rather than a QP problem per iteration and enjoys the favorable property that all iterates remain feasible with respect to (3). These two properties are also shared in a feasible equation-based method recently proposed by Kanzow [18]. We note that all of these methods are of the line-search type, and the superlinear/quadratic convergence

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

of these methods when applied to (3) requires that all the elements in $\partial\Phi(w^*)$ be nonsingular, where w^* is a solution of (1). Such a solution is usually called a strongly regular solution of (1).

There are several semismooth equation-based trust region methods which can be used to solve (1). The unconstrained trust region methods in [16, 20] are applicable to (2), while the box-constrained trust region method in [37] can be adapted to solve (3). Each of these methods requires at each iteration either an exact solution of the trust region subproblem [16]; a solution restricted to a (very small) subspace [20]; or an inexact solution which should satisfy some prescribed accuracy [37]. To be more precise, for global convergence only, the subspace in [20] can be as small as one-dimensional (i.e., spanned by the gradient direction), while for Ulbrich's method, any inexact solution satisfying the fraction of Cauchy decrease condition is enough (i.e., the affinely scaled gradient is a candidate). For local convergence, [20] requires that the subspace contain the generalized Newton direction, whereas [37] requires one to use the inexact generalized Newton direction. The use of iterative methods such as the conjugate gradient (CG) method is attractive because, on the one hand, first direction used in the CG method is the gradient direction, and quite often (e.g., when the trust region radius is larger than the length of the generalized Newton direction) the CG method yields an inexact generalized Newton direction and hence often speeds up the convergence process; on the other hand, when the number of variables is large, it is cost effective by the CG method to solve the trust region subproblem approximately. The key issue of efficiently implementing the CG method is the *preconditioning*. Although it is understood that no single preconditioning is "best" for all conceivable types of matrices, we will use the symmetric successive overrelaxation (SSOR) preconditioner in our numerical experiments. We will discuss it more in our numerical implementations.

In this paper we study how to apply the truncated CG method to a trust region subproblem of (3) so as to keep the computational cost at a reasonable level, and we show how to merge the truncated CG method with the semismooth Newton method as the iterates of our trust region method approach a minimizer, so that the use of the truncated CG method does not slow down the fast convergence of the proposed trust region method. Another favorable consequence of using the truncated CG method is that, although the quadratic term in the subproblem is constructed with $d^T H_k^T H_k d$, the calculation process of an approximation to the solution of the subproblem works directly on the matrix H_k, not on the usually condensed matrix $H_k^T H_k$. In addition, all iterates of our method remain feasible with respect to the simple bounds in (3), and the trust region subproblem is in a reduced form. This latter property is essential for the success of the truncated CG method and is guaranteed by incorporating an active-set strategy into the proposed trust region method. Finally, the superlinear/quadratic convergence of the proposed method is established under the assumption of nonsingularity.

The truncated CG method was first used by Toint [36] and Steihaug [33] to solve the trust region subproblem for unconstrained optimization problems and is shown to be efficient, especially in large scale optimization. Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable. Then the usual trust region subproblem for the unconstrained optimization problem $\min_{x\in\mathbb{R}^n} f(x)$ is

$$\begin{array}{ll} \min & \phi(d) = g^T d + \frac{1}{2} d^T B d \\ \text{s.t.} & \|d\| \le \Delta, \end{array} \tag{5}$$

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

where $\Delta > 0$ is a trust region bound, $g \in \mathbb{R}^n$ is the gradient of the objective function f at the current iterate, and $B \in \mathbb{R}^{n\times n}$ is symmetric and is an approximation to the Hessian of $f(x)$. Although the truncated CG method is widely used in practice, it was only recently proved that it indeed provides a sufficient decrease in the objective function for the case of strict convexity. In fact, when B is positive definite, Yuan[1] proved in [38] that the reduction in the objective function by the truncated CG method is at least half of the reduction by the global minimizer in the trust region. However, this result may be invalid when the bound constraint $x + d \geq 0$ is preserved. This can be shown by the following example. Consider the strictly convex problem in $\mathbb{R}^2$: $\min_{x\geq 0}\ x_1 + .5x_1^2 - x_2 + .5x_2^2$. Obviously, $(0, 1)$ is the unique solution. The trust region subproblem at $x = (\varepsilon, \varepsilon)$ is

$$\begin{array}{ll} \min & (1+\varepsilon)d_1 + (-1+\varepsilon)d_2 + \dfrac{1}{2}d_1^2 + \dfrac{1}{2}d_2^2 \\ \text{s.t.} & \|d\| \leq \Delta,\ x + d \geq 0. \end{array}$$

The truncated CG method for solving this subproblem first generates a direction by ignoring the bound constraint and then takes a small enough step along this direction to ensure feasibility. Hence, the next iterate is $x^1 = (0, 2\varepsilon/(1+\varepsilon))$ ($\Delta = 1$ and $\varepsilon \in (0,1)$). We continue to build the trust region subproblem around x^1; this time, the truncated CG method results in a direction which immediately goes infeasible, leading to the zero steplength. In other words, the truncated CG method fails to solve the strictly convex problem. The reason is that very small components of x may result in a very small (even zero) steplength. One way to avoid the collapse of the steplength is to build the trust region subproblem only around those components of the current iterate which are relatively large enough, while paying a special attention to the smaller ones.

The paper realizes the above ideas with a trust region method, which is solved by a truncated CG method. The paper is organized as follows. Some background is summarized in the next section. The subproblem is derived in section 3. A truncated CG method for this subproblem is introduced in section 4. Our algorithm is presented in section 5. Global and local convergence results are established in sections 6 and 7, respectively. Numerical results on a subset of problems from the MCPLIB collection [7] are presented in section 8. Finally, some conclusions are drawn in section 9.

2. Mathematical background.

2.1. Notation. A function $G : \mathbb{R}^t \to \mathbb{R}^t$ is called a C^k function if it is k times continuously differentiable, and an LC^k function if it is a C^k function and its kth derivative is locally Lipschitz continuous everywhere. The Jacobian of a C^1 function G at a point $w \in \mathbb{R}^t$ is denoted by $G'(w)$, whereas $\nabla G(w)$ is the transposed Jacobian. This notation is consistent with our notation of a gradient vector $\nabla g(w)$ for a real-valued function $g : \mathbb{R}^t \to \mathbb{R}$ since we view $\nabla g(w)$ as a column vector.

If $M \in \mathbb{R}^{t\times t}$, $M = (m_{ij})$, is any given matrix and $I, J \subseteq \{1, \ldots, t\}$ are two subsets, then M_{IJ} denotes the $|I| \times |J|$ submatrix with elements $m_{ij}, i \in I, j \in J$. Similarly, $M_{.J}$ indicates the submatrix with elements $m_{ij}, i \in \{1, \ldots, t\}, j \in J$; i.e., we obtain $M_{.J}$ from M by removing all columns with indices $j \notin J$. Similar notation is used for subvectors. If $w = (x^T, y^T, z^T)^T \in \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m$, we often simplify our

[1]Yuan attributes to P. Tseng a slightly weaker result that the reduction of the objective function by the truncated CG method is at least 1/3 (instead of 1/2) of the reduction by the global minimizer. And Yuan's result for the positive definite case is generalized to the positive semidefinite case in [6].

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

notation and write $w = (x, y, z)$. All vector norms used in this paper are Euclidean norms, and matrix norms are the 2-norms of the matrices. For a given symmetric positive definite matrix C, a norm induced by C is defined by $\|x\|_C := \sqrt{x^T C x}$.

2.2. Properties of the reformulation. Our analysis will make frequent use of some properties on the generalized Jacobian $\partial\Phi(w)$ (in the sense of Clarke [4]). An explicit formula of calculating an element from $\partial\Phi(w)$ is given in [9]. We will discuss more about it in the section of numerical experiments. Although Φ itself is not continuously differentiable in general, its square norm Ψ is continuously differentiable and

$$\nabla\Psi(w) = H^T\Phi(w) \qquad \forall\ H \in \partial\Phi(w). \tag{6}$$

Another favorable property of the equation operator Φ is that the nonsingularity of its generalized Jacobian is ensured by Robinson's strong regularity condition. We formally state this result as the following. For the precise definition, some further characterizations, and sufficient conditions for the strong regularity, we refer the reader to Robinson [32] as well as to Liu [22].

PROPOSITION 2.1 (see [9]). *A point $w^* = (x^*, y^*, z^*) \in \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m$ is a strongly regular solution of* (1) *if and only if all elements in the generalized Jacobian $\partial\Phi(w^*)$ are nonsingular.*

The next property follows from the fact that Φ is a (strongly) semismooth operator under certain smoothness assumptions for F, h, and g; see, e.g., [29, 31, 25, 13].

PROPOSITION 2.2. *For any $w = (x, y, z) \in \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m$, we have*

$$\|\Phi(w + d) - \Phi(w) - Hd\| = o(\|d\|) \qquad \textit{for } d \to 0 \textit{ and } H \in \partial\Phi(w + d).$$

If F is an LC^1 mapping, and h and g are LC^2 mappings, then

$$\|\Phi(w + d) - \Phi(w) - Hd\| = O(\|d\|^2) \qquad \textit{for } d \to 0 \textit{ and } H \in \partial\Phi(w + d).$$

An immediate consequence of the strong regularity of w^* and the semismoothness of Φ is that the function value $\|\Phi(w)\|$ provides a local error bound near w^*; see, e.g., [29, 25].

PROPOSITION 2.3. *Assume that w^* is a strongly regular solution of* (1). *Then there are constants $c_1 > 0$ and $\delta_1 > 0$ such that*

$$\|\Phi(w)\| \geq c_1\|w - w^*\|$$

for all w with $\|w - w^*\| \leq \delta_1$.

3. Subproblem. Given a current iterate $w^k = (x^k, y^k, z^k) \in \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m$ with $z^k \geq 0$, a traditional choice of the trust region subproblem for (3) is

$$\begin{array}{ll} \min & \nabla\Psi(w^k)^T d + \dfrac{1}{2} d^T H_k^T H_k d \\ \text{s.t.} & \|d\| \leq \Delta_k,\ z^k + d_z \geq 0, \end{array}$$

where $H_k \in \partial\Phi(w^k)$ and Δ_k is the current trust region radius. In order to make the truncated CG method successful with this subproblem, small components, as we observed in the introduction, should be detected and not involved in this subproblem, and the matrix $H_k^T H_k$ should be regularized to be positive definite in order for our algorithm to be well defined.

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

To make the idea precise, we introduce three index sets,

$$\begin{aligned}\mathcal{I} &:= \{1, \dots, n\},\\ \mathcal{P} &:= \{n+1, \dots, n+p\},\\ \mathcal{J} &:= \{n+p+1, \dots, n+p+m\},\end{aligned}$$

where $\mathcal{I}$ denotes the index set for the variables x, $\mathcal{P}$ is the index set for the equality constraints and the variables y, and $\mathcal{J}$ is the index set for the inequality constraints and the variables z. For example, if $w = (x, y, z) \in \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m$ is any given vector, then $w_{\mathcal{I}} = x$, $w_{\mathcal{P}} = y$, and $w_{\mathcal{J}} = z$. We also stress that if $j \in \mathcal{J}$ or $J \subseteq \mathcal{J}$, then w_j is a component of the z-part of the vector w and w_J is a subvector of the z-part of w.

To detect the small components of z^k, we introduce at each iteration an indicator $\delta_k > 0$ and the index set

$$J_k := \{j \in \mathcal{J} \,|\, w_j^k \le \delta_k\}, \tag{7}$$

which contains all indices whose corresponding components in z^k are thought to be small. We shall give it special attention in our algorithm since the truncated CG method may fail depending on it. Let

$$\bar{J}_k := \mathcal{I} \cup \mathcal{P} \cup (\mathcal{J} \setminus J_k). \tag{8}$$

In order to make the matrix $H_k^T H_k$ positive definite, we need a continuous function $\rho : \mathbb{R} \to \mathbb{R}$ with the following properties: (i) $\rho(s) \ge 0$ for all $s \in \mathbb{R}$, and (ii) $\rho(s) = 0$ if and only if $s = 0$. Such a function is usually called a forcing function.

From now on, we will often abbreviate the gradient vector $\nabla\Psi(w^k)$ by g^k (in contrast to $g(x^k)$, which denotes the function value of the inequality constraints at the current point x^k, so there should be no ambiguity). Partition $H_k = (H^k_{.J_k}, H^k_{.\bar{J}_k})$. We build our trust region subproblem only around the components of $w^k_{\bar{J}_k}$; that is,

$$\begin{array}{ll}\min & (g^k_{\bar{J}_k})^T d_{\bar{J}_k} + \dfrac{1}{2} d^T_{\bar{J}_k} \left((H^k_{.\bar{J}_k})^T H^k_{.\bar{J}_k} + \rho(\Psi(w^k)) I \right) d_{\bar{J}_k} \\ \text{s.t.} & \|d_{\bar{J}_k}\| \le \Delta_k, \ w^k_{\mathcal{J}\setminus J_k} + d_{\mathcal{J}\setminus J_k} \ge 0.\end{array} \tag{9}$$

It is clear that the above subproblem is a strictly convex quadratic problem if $\Psi(w^k) \neq 0$.

4. Truncated CG method. In this section, we adapt the truncated preconditioned conjugate gradient (PCG) method described in [33] to our subproblem, which is a $q := (n + p + m - |J_k|)$-dimensional convex problem.

Suppose a symmetric positive definite matrix $C \in \mathbb{R}^{q\times q}$ is given with decomposition property $C = P^T P$, where P is nonsingular. For simplicity we denote the variable $d_{\bar{J}_k}$ by s. Consider the preconditioned version of (9):

$$\begin{array}{ll}\min_{s\in\mathbb{R}^q} & m_k(s) := s^T b + \dfrac{1}{2} s^T \mathcal{B}_k s \\ \text{s.t.} & \|s\|_C \le \Delta_k, \ w^k_{\mathcal{J}\setminus J_k} + s_{\mathcal{J}\setminus J_k} \ge 0,\end{array} \tag{10}$$

where $b := g^k_{\bar{J}_k}$, $A := H^k_{.\bar{J}_k}$, $\sigma := \rho(\Psi(w^k))$, and $\mathcal{B}_k := A^T A + \sigma I$. Taking into consideration the special structure of $\mathcal{B}_k$ and the decomposition property of the preconditioner C, we arrive at the following truncated PCG method for problem (10).

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

ALGORITHM 4.1 (truncated PCG method).

```
(S.0) Let $s^0 = 0,\ r^0 = b,\ \tilde{r}^0 = P^{-T}r^0,\ p^0 = -\tilde{r}^0,\ i := 0$.
(S.1) If $\|\nabla m_k(s^i)\| = 0$, then set $s^* = s^i$ and go to (S.4). Otherwise calculate
          $t^i = P^{-1}p^i,\ q^i = At^i$, and $\alpha_i = \|\tilde{r}^i\|^2/(\|q^i\|^2 + \sigma\|t^i\|^2)$.
(S.2) If $\|s^i + \alpha_i t^i\|_C \geq \Delta_k$, then go to (S.3). Otherwise set
          $s^{i+1} := s^i + \alpha_i t^i,\ r^{i+1} := r^i + \alpha_i(\sigma t^i + A^T q^i)$,
          $\tilde{r}^{i+1} := P^{-T}r^{i+1},\ \beta_i := \|\tilde{r}^{i+1}\|^2/\|\tilde{r}^i\|^2,\ p^{i+1} := -\tilde{r}^{i+1} + \beta_i p^i$.
      Set $i := i + 1$, and go to (S.1).
(S.3) Calculate $\alpha_i^* \geq 0$ satisfying $\|s^i + \alpha_i^* t^i\|_C = \Delta_k$; set $s^* := s^i + \alpha_i^* t^i$.
(S.4) Compute the largest $\tau_k \geq 0$ satisfying $w^k_{\mathcal{J}\setminus J_k} + \tau s^*_{\mathcal{J}\setminus J_k} \geq 0$ for all $\tau \in (0, \tau_k]$.
(S.5) Output the approximate solution to (10): $d^k_{\bar{J}_k} = \min\{1, \tau_k\}s^*$.
```

Remarks. First, we note that the computation of s^* above is exactly Steihaug's algorithm [33]. Second, in each iteration of the PCG method, there are two matrix-vector multiplications involving A (to get At^i and $A^T(At^i)$) and two matrix-inverse multiplications involving P (to get $P^{-T}r^i$ and $P^{-1}(P^{-T}r^i)$). If we choose C to be the SSOR preconditioner, we shall see in section 8 by referring to several specific references that the usually condensed matrix $\mathcal{B}_k$ is not involved in the calculation. In fact, only nonzero elements in A come to be used. Third, the termination rule in (S.1) is only for theoretical purposes. We shall use a more practical criterion in our implementation.

The vector s^* generated above has a close relation to the exact solution of the following problem:

$$\min\ m_k(d_{\bar{J}_k}) \qquad \text{s.t. } \|d_{\bar{J}_k}\|_C \leq \Delta_k. \tag{11}$$

This relation follows from a recent result of Yuan [38, Thm. 2].

PROPOSITION 4.2. *Suppose that $\mathcal{B}_k$ is positive definite. Let $d^*_{\bar{J}_k}$ be the exact solution of* (11) *and s^* be generated by Algorithm* 4.1. *Then we have*

$$m_k(s^*) \leq \frac{1}{2} m_k(d^*_{\bar{J}_k}).$$

Moreover, if $\|d^*_{\bar{J}_k}\|_C < \Delta_k$, then $s^* = d^*_{\bar{J}_k}$.

Proof. As remarked above, the calculation of s^* in Algorithm 4.1 is actually Steihaug's algorithm applied to the subproblem (11). Let $y = Ps$ and $y^* = Ps^*$. Then y^* is the point obtained by applying the truncated CG method to the following trust region problem:

$$\begin{array}{ll} \min_{y\in\mathbb{R}^q} & \tilde{m}_k(y) := y^T(P^{-T}b) + \dfrac{1}{2}y^T(P^{-T}\mathcal{B}_k P^{-1})y \\ \text{s.t.} & \|y\| \leq \Delta_k. \end{array} \tag{12}$$

Let $\tilde{y}^*$ be the unique solution of (12). According to a result of Yuan [38, Thm. 2], it holds that $\tilde{m}_k(y^*) \leq \frac{1}{2}\tilde{m}_k(\tilde{y}^*)$ and $y^* = \tilde{y}^*$ if $\|\tilde{y}^*\| < \Delta_k$. We note that $\tilde{m}_k(y^*) = m_k(s^*)$ and $\tilde{m}_k(\tilde{y}^*) = m_k(d^*_{\bar{J}_k})$. Then the inequality relation in the proposition follows. Moreover, $s^* = d^*_{\bar{J}_k}$ if $\|d^*_{\bar{J}_k}\|_C < \Delta_k$. $\square$

In the following as well as in our convergence analysis, we assume that $C_k = I$ for simplicity, where C_k is the preconditioner in (10) at each iteration. However, to keep

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

all convergence results in sections 6 and 7 valid, we need more assumptions on the preconditioner sequence $\{C_k\}$; namely, there exist two constants $\underline{\kappa}$ and $\bar{\kappa}$ such that, for all k,

$$\|C_k^{-1}\| \le \underline{\kappa} \quad \text{and} \quad \|C_k\| \le \bar{\kappa}. \tag{13}$$

This condition implies that C_k has a condition number that is bounded independent of the iterate. The latter condition has been singled out in [21, p. 1120] to emphasize its theoretical importance in a trust region method. We note that the two conditions are equivalent under boundedness of the whole iterate sequence $\{w^k\}$. Standard convergence proofs involving condition (13) (in the special case that C_k is diagonal) can be found in [5] (proofs leading up to Theorem 11 there). We also note that we use the Hessian matrix H_k in building up $\mathcal{B}_k$, implying that the standard condition restricted on $\mathcal{B}_k$ in trust region methods is fulfilled automatically in our setting. Proposition 4.2 allows us to put a bound on the predicted decrease $m_k(d^k_{\bar{J}_k})$. Two bounds are given in the following result. The first corresponds to the case where $\tau_k \ge 1$ so that $d^k_{\bar{J}_k} = s^*$; the second corresponds to the case where $\tau_k < 1$ so that $d^k_{\bar{J}_k} \ne s^*$.

PROPOSITION 4.3. *Let $d^k_{\bar{J}_k}$ and s^* be generated by Algorithm* 4.1 *applied to* (9), *and define*

$$\Omega_k := \{j \in \mathcal{J} \setminus J_k | -s_j^* > w_j^k\}.$$

Then the following statements hold:

(i) *If Ω_k is empty (in particular if $\delta_k \ge \Delta_k$), then we have*

$$m_k(d^k_{\bar{J}_k}) \le -\frac{1}{4}\|g^k_{\bar{J}_k}\| \min\left\{\Delta_k, \frac{\|g^k_{\bar{J}_k}\|}{\|\mathcal{B}_k\|}\right\}.$$

(ii) *If $\Omega_k \ne \emptyset$, then*

$$m_k(d^k_{\bar{J}_k}) \le -\frac{\delta_k}{4\Delta_k}\|g^k_{\bar{J}_k}\| \min\left\{\Delta_k, \frac{\|g^k_{\bar{J}_k}\|}{\|\mathcal{B}_k\|}\right\}.$$

Proof. (i) Let d^* be the unique solution to (9). Then it follows from [27, Thm. 4] that

$$m_k(d^*) \le -\frac{1}{2}\|g^k_{\bar{J}_k}\| \min\left\{\Delta_k, \frac{\|g^k_{\bar{J}_k}\|}{\|\mathcal{B}_k\|}\right\}. \tag{14}$$

Also by simple calculation we have $\tau_k \ge 1$ if $\Omega_k = \emptyset$. This is also true in particular if $\delta_k \ge \Delta_k$. Hence $d^k_{\bar{J}_k} = s^*$ is also generated by Yuan's truncated CG method [38] applied to (9) with the simple constraints not being violated. Therefore, Proposition 4.2 implies

$$m_k(d^k_{\bar{J}_k}) = m_k(s^*) \le \frac{1}{2} m_k(d^*). \tag{15}$$

The combination of (15) and (14) gives the result in (i).

(ii) Let $j \in \Omega_k$. Then $\tau_k < 1$ and

$$\Delta_k \ge -s_j^* > w_j^k > \delta_k.$$

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Hence

$$d^k_{\bar{J}_k} = \tau_k s^* \quad \text{and} \quad \tau_k \geq \delta_k/\Delta_k.$$

Now we consider the one-dimensional function

$$\mathcal{M}(t) := m_k(ts^*) = t(g^k_{\bar{J}_k})^T s^* + \frac{1}{2}t^2(s^*)^T\mathcal{B}_k s^*.$$

Let

$$\tau_* = \frac{-(g^k_{\bar{J}_k})^T s^*}{(s^*)^T\mathcal{B}_k s^*}.$$

It follows from the fact that $\mathcal{M}(1) = m_k(s^*) \leq 0$ that $2\tau_* \geq 1$. Now we consider two cases. First if $\tau_k \leq \tau_*$, the convexity of $\mathcal{M}$ implies that

$$\begin{aligned} m_k(d^k_{\bar{J}_k}) &= \mathcal{M}(\tau_k) \leq \mathcal{M}(\delta_k/\Delta_k) \\ &= \frac{\delta_k}{\Delta_k}(g^k_{\bar{J}_k})^T s^* + \frac{1}{2}\left(\frac{\delta_k}{\Delta_k}\right)^2 (s^*)^T\mathcal{B}_k s^* \\ &\leq \frac{\delta_k}{\Delta_k}\mathcal{M}(1) = \frac{\delta_k}{\Delta_k} m_k(s^*). \end{aligned} \tag{16}$$

If $\tau_k \geq \tau_*$, then it is easy to see from the convexity of $\mathcal{M}$ again that

$$m_k(d^k_{\bar{J}_k}) = \mathcal{M}(\tau_k) \leq \mathcal{M}(1) = m_k(s^*). \tag{17}$$

Now the result in (ii) follows from (14), (16), (17), and Proposition 4.2. ∎

5. Algorithm. Suppose $\Delta_k, w^k, J_k, \bar{J}_k$, and the function $m_k(\cdot)$ are given as in the last section, and a search direction $\tilde{d}^k$ is partitioned as

$$\tilde{d}^k = \begin{pmatrix} \tilde{d}^k_{J_k} \\ \tilde{d}^k_{\bar{J}_k} \end{pmatrix}.$$

Let the ratio between the actual decrease and the predicted decrease associated with the direction $\tilde{d}^k$ be calculated by

$$r_k := \left(\Psi(w^k) - \Psi(w^k + \tilde{d}^k)\right)/\text{Pred}_k, \tag{18}$$

where

$$\text{Pred}_k := -(g^k_{J_k})^T \tilde{d}^k_{J_k} - m_k(\tilde{d}^k_{\bar{J}_k}).$$

The update rule used in our trust region algorithm is as follows:

$$w^{k+1} := \begin{cases} w^k & \text{if } r_k < \rho_1, \\ w^k + \tilde{d}^k & \text{if } r_k \geq \rho_1, \end{cases} \qquad \Delta_{k+1} := \begin{cases} \sigma_1\Delta_k & \text{if } r_k < \rho_1, \\ \max\{\Delta_{\min}, \Delta_k\} & \text{if } r_k \in [\rho_1, \rho_2), \\ \max\{\Delta_{\min}, \sigma_2\Delta_k\} & \text{if } r_k \geq \rho_2, \end{cases} \tag{19}$$

where $\Delta_{\min} > 0$ is a prescribed constant. The proposed trust region algorithm is then formally stated below.

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Algorithm 5.1 (trust region algorithm).

(S.0) *Choose* $w^0 = (x^0, y^0, z^0) \in \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m$ *with* $z^0 \geq 0$, $\Delta_0 > 0$, $0 < \rho_1 < \rho_2 < 1$, $0 < \sigma_1 < 1 < \sigma_2$, $\Delta_{\min} > 0$, $\gamma \in (0,1), c > 0$, $\delta > 0$, $\epsilon > 0$, $\mathrm{ind}_0 := 0$, $\beta_0 = 0$, *and set* $k := 0$.

(S.1) *Let*

$$\delta_k := \min\left\{\delta, c\sqrt{\|\Phi(w^k)\|}\right\}$$

and define the set J_k *and* $\bar{J}_k$ *by* (7) *and* (8), *respectively.*

(S.2) *Let*

$$v^k := \begin{pmatrix} v^k_{J_k} \\ v^k_{\bar{J}_k} \end{pmatrix},$$

where

$$v^k_{J_k} = \min\{w^k_{J_k}, g^k_{J_k}\} \quad \textit{and} \quad v^k_{\bar{J}_k} = g^k_{\bar{J}_k}.$$

If $\|v^k\| \leq \epsilon$, *stop.*

(S.3) *Choose preconditioner* C_k *and let* $d^k_{\bar{J}_k}$ *be the final iterate of Algorithm* 4.1 *for the trust region subproblem* (10).

(S.4) *Compute the search directions*

$$d^k := \begin{pmatrix} -w^k_{J_k} \\ d^k_{\bar{J}_k} \end{pmatrix} \quad \textit{and} \quad \tilde{d}^k := \begin{pmatrix} -\min\{1, \Delta_k\} v^k_{J_k} \\ d^k_{\bar{J}_k} \end{pmatrix}.$$

(S.5) (i) *If* $\mathrm{ind}_k = 0$, *check if the following rule holds:*

$$\Psi(w^k + d^k) \leq \gamma\sqrt{\|\Phi(w^k)\|}. \tag{20}$$

If test (20) *is successful, then let*

$$w^{k+1} := w^k + d^k, \quad \Delta_{k+1} := \max\{\Delta_{\min}, \sigma_2\Delta_k\}, \quad \gamma_{k+1} := \frac{\Psi(w^{k+1})}{\Psi(w^k)}$$

and

$$\bar{\gamma} := \gamma_{k+1} \textit{ if } \gamma_{k+1} \geq \gamma, \quad \beta_{k+1} := \begin{cases} \Psi(w^{k+1}) & \textit{if } \gamma_{k+1} \geq \gamma, \\ \beta_k & \textit{if } \gamma_{k+1} < \gamma, \end{cases}$$

$$\mathrm{ind}_{k+1} := \begin{cases} 1 & \textit{if } \gamma_{k+1} \geq \gamma, \\ 0 & \textit{if } \gamma_{k+1} < \gamma. \end{cases}$$

If test (20) *is not successful, then calculate* r_k *by* (18), *update* w^{k+1} *and* Δ_{k+1} *according to rule* (19), *and let* $\beta_{k+1} := \beta_k$, $\mathrm{ind}_{k+1} := 0$.

(ii) *If* $\mathrm{ind}_k = 1$, *check if the following holds:*

$$\Psi(w^k + d^k) \leq \frac{\gamma}{\bar{\gamma}}\beta_k. \tag{21}$$

If test (21) *is successful, let*

$$w^{k+1} := w^k + d^k, \ \Delta_{k+1} := \max\{\Delta_{\min}, \sigma_2\Delta_k\}, \ \beta_{k+1} := \beta_k, \ \mathrm{ind}_{k+1} := 0.$$

If test (21) *is not successful, then calculate* r_k *by* (18), *update* w^{k+1} *and* Δ_{k+1} *according to the rule* (19), *and let* $\beta_{k+1} := \beta_k$, $\mathrm{ind}_{k+1} := 1$.

(S.6) *Set* $k := k + 1$ *and go to* (S.1).

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

More explanation on Algorithm 5.1 is as follows. The indicator δ_k defined in (S.1) was also used in [19, 18] to identify the actual active set under the nonsingularity assumption. A result due to Kanzow and Qi [19] justifies the termination rule in (S.2). We will present this result in a lemma below. The direction $d^k_{\bar{J}_k}$ in (S.3) has been extensively discussed in the last section. The crucial parts of Algorithm 5.1 are (S.4) and (S.5). In (S.4) two directions are defined. The direction $\tilde{d}^k$ (which we call safe step below) is always a descent direction of the function $\Psi(\cdot)$ at w^k if Δ_k is sufficiently small; while direction d^k (fast step) will yield a superlinear decrease in the function value of $\Psi(\cdot)$ when w^k is sufficiently close to a strongly regular solution w^*. The task in (S.5) is then to decide which step we should take. Fast steps are accepted in a nonmonotone fashion, so that it can happen that an accepted fast step increases the value of Ψ. To keep control over those possible increases, a flag `ind` is used to distinguish between two states of the algorithm: If `ind` $= 0$ (which is the case at the very beginning), the fast step d^k is accepted if test (20) is successful. Now, if $\Psi(w^k + d^k) \geq \gamma\Psi(w^k)$, the flag `ind` is set to 1 (and remains raised until it is cleared again) to signal that d^k did not achieve sufficient decrease. Now consider any iteration k that is entered with `ind` $= 1$, indicating that the most recent accepted fast step d^l $(l < k)$ did not achieve sufficient decrease. In this situation, the fast step d^k is accepted only if test (21) is successful. If this occurs, `ind` is set to 0 again. In all iterations where the fast step is not accepted, the safe step is used as the trial step of the trust region method with a standard reduction-ratio-based acceptance test. We stress that both $\bar{\gamma}$ and β_k are used to record the cases where (20) is successful and $\gamma_{k+1} \geq \gamma$, i.e., to record the cases where the function values are possibly increased.

From now on we assume that $\epsilon = 0$. The following result from [19, Lem. 1] justifies the termination criterion used in our trust region algorithm. We recall that a point $w^* = (x^*, y^*, z^*) \in \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m$ with $z^* \geq 0$ is a stationary point of problem (3) if $\nabla_x \Psi(w^*) = 0$, $\nabla_y \Psi(w^*) = 0$, and

$$z_i^* > 0 \Longrightarrow \frac{\partial \Psi(w^*)}{\partial z_i} = 0, \quad z_i^* = 0 \Longrightarrow \frac{\partial \Psi(w^*)}{\partial z_i} \geq 0.$$

LEMMA 5.2. *Let $w^k = (x^k, y^k, z^k) \in \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m$ be any given point with $z^k \geq 0$. Then the following holds:*

$$w^k \textit{ is a stationary point of } (3) \quad \Longleftrightarrow \quad v^k = 0 \quad \Longleftrightarrow \quad (g^k)^T v^k = 0.$$

The following result shows that the iterates $\{w^k\}$ generated by Algorithm 5.1 stay feasible with respect to the simple bounds in (3).

LEMMA 5.3. *Let $w^k = (x^k, y^k, z^k) \in \mathbb{R}^n \times \mathbb{R}^p \times \mathbb{R}^m$ be any given point with $z^k \geq 0$, and assume that w^k is not a stationary point of* (3). *Then the next iterate w^{k+1} can be computed by Algorithm* 5.1 *and it holds that $z^{k+1} \geq 0$.*

Proof. Let w^k be given as in Lemma 5.3. We have no problem running steps (S.1)–(S.4) of Algorithm 5.1. Hence, for Algorithm 5.1 to be well defined, we need to show that (S.5) is well defined

Since w^k is not a stationary point of (3), $\Psi(w^k) > 0$ so that $\bar{\gamma}$ in (S.5)(i) is well defined if (20) is successful. If (20) does not hold, we need only to show $\text{Pred}_k \neq 0$ so that r_k is well defined. It follows from Proposition 4.3 that

$$m_k(d^k_{\bar{J}_k}) \leq 0 \quad \text{ and } \quad m_k(d^k_{\bar{J}_k}) = 0 \iff g^k_{\bar{J}_k} = 0.$$

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

On the other hand we have, for $j \in J_k$,

$$g_j^k v_j^k = \begin{cases} (g_j^k)^2 & \text{if } g_j^k \le w_j^k, \\ g_j^k w_j^k & \text{if } g_j^k > w_j^k. \end{cases}$$

Noting that $w_j^k \ge 0$ for $j \in J_k$, we obtain

$$(g_{J_k}^k)^T v_{J_k}^k \ge 0.$$

Hence $\mathrm{Pred}_k = \min\{1, \Delta_k\}(g_{J_k}^k)^T v_{J_k}^k - m_k(d_{\bar{J}_k}^k) \ge 0$, and if $\mathrm{Pred}_k = 0$, we must have both $(g_{J_k}^k)^T v_{J_k}^k = 0$ and $g_{\bar{J}_k}^k = 0$, which means that $(g^k)^T v^k = (g_{J_k}^k)^T v_{J_k}^k + \|g_{\bar{J}_k}^k\|^2 = 0$. By Lemma 5.2, w^k must be a stationary point of (3), contradicting the assumption of this lemma. Hence $\mathrm{Pred}_k > 0$, and consequently r_k is well defined and w^{k+1} is obtained.

Now we prove $z^{k+1} \ge 0$. There are three possible ways to determine w^{k+1}, namely, $w^{k+1} = w^k$, $w^{k+1} = w^k + d^k$, or $w^{k+1} = w^k + \tilde{d}^k$. The first case is trivial, so we consider the remaining two cases. We note that the components w_j^{k+1}, $j \in \mathcal{J} \setminus J_k$, are updated by

$$w_j^{k+1} = w_j^k + (d_{\bar{J}_k}^k)_j = w_j^k + \min\{1, \tau_k\} s_j^* \ge \begin{cases} w_j^k & \text{if } s_j^* \ge 0, \\ w_j^k + \tau_k s_j^* & \text{if } s_j^* < 0, \end{cases}$$

where s^* is computed by Algorithm 4.1. It follows from (S.4) of Algorithm 4.1 that $w_j^{k+1} \ge 0$. For the components j belonging to J_k, if $w^{k+1} = w^k + d^k$, then $w_j^{k+1} = w_j^k - w_j^k = 0$; if $w^{k+1} = w^k + \tilde{d}^k$, then

$$\begin{aligned} w_j^{k+1} &= w_j^k - \min\{1, \Delta_k\} \min\{w_j^k, g_j^k\} \\ &\ge \begin{cases} w_j^k \ge 0 & \text{if } \min\{w_j^k, g_j^k\} \le 0, \\ w_j^k - w_j^k = 0 & \text{if } \min\{w_j^k, g_j^k\} > 0. \end{cases} \end{aligned}$$

This proves that $w_j^{k+1} \ge 0$ for all $j \in \mathcal{J}$. $\square$

We note that as long as w^k is not a global minimizer of (3), Algorithm 4.1 is always successful for subproblem (9), and the estimation in Proposition 4.3 always holds since $\mathcal{B}_k$ is always positive definite. So we can apply an induction argument by invoking Lemma 5.3 and then obtain the following result.

THEOREM 5.4. *Algorithm* 5.1 *is well defined and generates a sequence* $\{w^k\} = \{(x^k, y^k, z^k)\}$ *with* $z^k \ge 0$ *for all* k.

6. Global convergence.

From now on we assume that Algorithm 5.1 generates an infinite sequence $\{w^k\}$. Let K contain all the indices at which the function value is possibly increased; that is,

$$K := \left\{ k \in \{0, 1, 2, \ldots\} \mid \mathrm{ind}_k = 0,\ \Psi(w^k + d^k) \le \gamma \sqrt{\|\Phi(w^k)\|} \text{ and } \gamma_{k+1} \ge \gamma \right\}. \tag{22}$$

Then we have the following convergence result.

LEMMA 6.1. *Suppose that* K *contains infinitely many iterations. Then*

$$\lim_{k \to \infty} \Psi(w^k) = 0.$$

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Hence, every limit point of $\{w^k\}$ *is a solution of* (1) *and therefore a stationary point of* (3).

Proof. Let us denote K by

$$K = \{k_0, k_1, k_2, \ldots\}.$$

In the following we want to prove

(a) the sequence $\{\Psi(w^k)\}_{k\in K}$ converges to zero, i.e., $\lim_{l\to\infty}\Psi(w^{k_l}) = 0$;

(b) the sequence $\{\Psi(w^k)\}_{(k-1)\in K}$ converges to zero, i.e., $\lim_{l\to\infty}\Psi(w^{k_l+1}) = 0$; and

(c) for any k such that $k_l + 1 < k \le k_{l+1}$ for some $l \in \{0, 1, \ldots\}$, we have

$$\Psi(w^k) \le \Psi(w^{k_l+1}). \tag{23}$$

It is easy to see that (b) follows directly from (a) since $\Psi(w^{k_l+1}) \le \gamma\sqrt{\|\Phi(w^{k_l})\|}$. (b) and (c) together yield

$$\lim_{\substack{k\in(k_l+1,k_{l+1}]\\ k\to\infty}} \Psi(w^k) = 0.$$

We observe that every iterate w^k must belong to $k \in K$, or $(k-1) \in K$, or $k_l + 1 < k \le k_{l+1}$ for some $l \in \{0, 1, 2, \ldots\}$. Hence we must have $\lim_{k\to\infty}\Psi(w^k) = 0$ if (a), (b), and (c) are true. Consequently, every limit of $\{w^k\}$ is a solution of (1) and therefore a stationary point of (3).

Now we prove (a) and (c) together. First, for any k between 0 and k_0, i.e., $0 \le k < k_0$, we have $\mathrm{ind}_k = 0$. This means Algorithm 5.1 uses (S.5)(i) to find the next iterate. If (20) is successful at k, then we must have $\gamma_{k+1} < \gamma$ (otherwise k would belong to K, resulting in $k_0 \le k$, a contradiction). Hence it follows from the definition of γ_{k+1} that

$$\Psi(w^{k+1}) = \gamma_{k+1}\Psi(w^k) < \gamma\Psi(w^k) < \Psi(w^k). \tag{24}$$

If (20) is not successful at k, then w^{k+1} is obtained by rule (19). In this case, it is obvious that

$$\Psi(w^{k+1}) \le \Psi(w^k). \tag{25}$$

By the induction argument on k between 0 and k_0, relations (24) and (25) give us that

$$\Psi(w^{k_0}) \le \Psi(w^{k_0-1}) \le \cdots \le \Psi(w^1) \le \Psi(w^0). \tag{26}$$

Now we take a look at how Algorithm 5.1 runs at iterations between k_l and k_{l+1}. Our first observation is that

$$\beta_{k_{l+1}} = \beta_{k_{l+1}-1} = \cdots = \beta_{k_l+1} = \Psi(w^{k_l+1}). \tag{27}$$

For any $k_l \in K$, according to (S.5)(i)

$$\Psi(w^{k_l+1}) = \bar\gamma\Psi(w^{k_l}), \quad \mathrm{ind}_{k_l+1} = 1 \text{ (since } \gamma_{k_l+1} \ge \gamma). \tag{28}$$

Then Algorithm 5.1 uses (S.5)(ii) (since $\mathrm{ind}_{k_l+1} = 1$) to generate the next iterate w^{k_l+2}. The algorithm will repeat (S.5)(ii) until (21) is successful at some iterate, say

$w^{\bar{k}}$, putting $\text{ind}_{\bar{k}+1}$ back to *zero* so that the algorithm uses (S.5)(i) to find the next iterate until k reaches k_{l+1}. Hence, there is exactly one such $\bar{k}$ satisfying $k_l + 1 \le \bar{k} < k_{l+1}$. At iteration k, where $k_l + 1 \le k < \bar{k}$, the algorithm always uses rule (19) to generate the next iterate, which means that

$$\Psi(w^{k_l+1}) \ge \Psi(w^{k_l+2}) \ge \cdots \ge \Psi(w^{\bar{k}}). \tag{29}$$

At this stage, $\bar{\gamma}$ remains unchanged; i.e., $\bar{\gamma} = \Psi(w^{k_l+1})/\Psi(w^{k_l})$ for all $k \in [k_l + 1, \bar{k}]$. Since (21) is successful at $\bar{k}$, we have

$$\begin{aligned} \Psi(w^{\bar{k}+1}) &\le \frac{\gamma}{\bar{\gamma}}\beta_{\bar{k}} = \frac{\gamma}{\bar{\gamma}}\Psi(w^{k_l+1}) \qquad \text{(using (27))} \\ &\le \gamma \frac{\Psi(w^{k_l})}{\Psi(w^{k_l+1})}\Psi(w^{k_l+1}) \qquad (\text{using } \bar{\gamma} = \Psi(w^{k_l+1})/\Psi(w^{k_l})) \\ &= \gamma\Psi(w^{k_l}). \end{aligned} \tag{30}$$

On the other hand, at iteration k, where $\bar{k} + 1 \le k < k_{l+1}$, the algorithm uses either rule (19) or (20) to generate the next iterate. If the algorithm uses (19), then it is obvious that $\Psi(w^{k+1}) \le \Psi(w^k)$. If the algorithm uses (20), then we must have $\gamma_{k+1} < \gamma$, which also yields $\Psi(w^{k+1}) < \Psi(w^k)$. Hence, we have

$$\Psi(w^{\bar{k}+1}) \ge \Psi(w^{\bar{k}+2}) \ge \cdots \ge \Psi(w^{k_{l+1}}). \tag{31}$$

Putting (30) and (31) together, we obtain by an induction argument

$$\Psi(w^{k_{l+1}}) \le \gamma\Psi(w^{k_l}) \le \gamma^2\Psi(w^{k_{l-1}}) \le \cdots \le \gamma^{l+1}\Psi(w^0).$$

The last inequality uses (26). Taking the limit in the above inequalities gives (a). Finally, it follows from (30) that

$$\Psi(w^{\bar{k}+1}) \le \gamma\Psi(w^{k_l}) \le \gamma_{k_l+1}\Psi(w^{k_l}) = \Psi(w^{k_l+1}).$$

This together with (29) and (31) implies (23). ☐

We now consider the case that K contains only finitely many elements, say

$$K = \{k_0, k_1, \ldots, k_l\}.$$

LEMMA 6.2. *Suppose that K contains finitely many elements. Then the following hold:*

(i) *The sequence $\{\Psi(w^k)\}_{k \ge k_l+1}$ is monotonically decreasing.*

(ii) *If test (20) holds infinitely many times, then*

$$\lim_{k\to\infty} \Psi(w^k) = 0.$$

In this case, every limit of $\{w^k\}$ is a solution of (1).

Proof. Since k_l is the last element in K, we have by the definition of K that $\text{ind}_{k_l+1} = 1$, $\bar{\gamma} = \Psi(w^{k_l+1})/\Psi(w^{k_l})$, and $\bar{\gamma}$ remains unchanged from k_l+1 and onward.

(i) Since $\text{ind}_{k_l+1} = 1$, the algorithm uses (S.5)(ii) to generate the iterate w^{k_l+2}. We note that test (21) could possibly hold only once after the iteration $k_l + 1$ since once (21) holds the algorithm puts ind_k back to *zero* and will never use (S.5)(ii) thereafter. Suppose that (21) holds at iteration $\bar{k}$ ($\bar{k} \ge k_l + 1$). For iterations k

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

satisfying $k_l + 1 \le k < \bar{k}$, $\text{ind}_k = 1$ and hence the algorithm uses rule (19) to update w^k. Therefore, we have

$$\Psi(w^{k+1}) \le \Psi(w^k) \qquad \forall\, k \in [k_l + 1, \bar{k}).$$

At iteration $\bar{k}$, we have, from $\bar{\gamma} \ge \gamma$,

$$\Psi(w^{\bar{k}+1}) \le \frac{\gamma}{\bar{\gamma}} \Psi(w^{\bar{k}}) \le \Psi(w^{\bar{k}})$$

and $\text{ind}_{\bar{k}+1} = 0$. So from iteration $\bar{k}+1$ onward, the algorithm uses (S.5)(i) to update w^k. Let $k \ge \bar{k} + 1$ be given. If (20) is successful at k, then we must have $\gamma_{k+1} < \gamma$ (otherwise $k \in K$, resulting in $k \le k_l$, a contradiction of $k \ge \bar{k} + 1 \ge k_l + 2$). Then

$$\Psi(w^{k+1}) = \gamma_{k+1}\Psi(w^k) < \gamma\Psi(w^k). \tag{32}$$

If (20) is not successful at k, the algorithm uses rule (19) to update w^k, giving $\Psi(w^{k+1}) \le \Psi(w^k)$. If (21) never holds from $k_l + 1$ onward, then the algorithm uses rule (19) to generate w^{k+1} for all $k \ge k_l + 1$. We then have $\Psi(w^{k+1}) \le \Psi(w^k)$ for all $k \ge k_l + 1$. All in all, we have proved the statement in (i).

(ii) Suppose that test (20) holds infinitely many times, which means that there exists $\bar{k} \ge k_l + 1$ such that (21) holds at $\bar{k}$. The algorithm uses (S.1)(i) to update w^k from $\bar{k} + 1$ onward, and (20) holds infinitely many times after $\bar{k} + 1$. Hence the relation (32) holds infinitely many times after $\bar{k} + 1$. Noting that $\{\Psi(w^k)\}_{k \ge k_l + 1}$ is monotonically decreasing, we certainly have (ii) from (32). ∎

The goal we want to achieve in this section is that any limit of the sequence $\{w^k\}$ is a stationary point of (3), which under reasonable conditions [11, Thm. 3.1] is already a solution of (1). Because of Lemmas 6.1 and 6.2, we need only consider the case that K contains finitely many elements and test (20) holds only finitely many times. In other words, we need only consider the case that Algorithm 5.1, after finitely many iterations, uses only rule (19) to update w^k. Without loss of generality we assume from now on that the whole sequence $\{w^k\}$ is generated according to the trust region rule (19). The convergence analysis for this part is quite standard from the trust region point of view.

LEMMA 6.3. *Suppose that the whole sequence $\{w^k\}$ was generated according to rule* (19), *and that w^* is the limit of a subsequence $\{w^k\}_{\tilde{K}}$. If w^* is not a stationary point of* (3), *then*

$$\lim_{k\to\infty} \inf_{k \in \tilde{K}} \Delta_k > 0.$$

Proof. It is obvious that the function value sequence $\{\Psi(w^k)\}_{k\ge 1}$ is monotonically decreasing, and so is the sequence $\{\delta_k\}$. Moreover,

$$\lim_{k\to\infty} \delta_k = \delta_* := \min\{\delta, c\sqrt{\|\Phi(w^*)\|}\} > 0;$$

the last inequality uses the fact $\Psi(w^*) > 0$ as w^* is not a stationary point of (3). Now define the index set

$$\bar{K} := \{k - 1 |\ k \in \tilde{K}\}.$$

Then the subsequence $\{w^{k+1}\}_{k\in\bar{K}}$ converges to w^*. Suppose that the result of this lemma does not hold. Subsequencing if necessary we can assume that

$$\lim_{k\to\infty, k\in\bar{K}} \Delta_{k+1} = 0. \tag{33}$$

In view of the updating rule for the trust region radius (note that the lower bound

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

$\Delta_{\min} > 0$ plays an important role here), (33) implies that for all iterations $k \in \bar{K}$ sufficiently large, we have

$$r_k < \rho_1, \qquad w^k = w^{k+1}, \qquad \Delta_{k+1} = \sigma_1 \Delta_k. \tag{34}$$

Hence

$$\{w^k\}_{\bar{K}} \to w^* \quad \text{and} \quad \lim_{k\to\infty, k\in\bar{K}} \Delta_k = 0. \tag{35}$$

Because of the continuity of $\nabla\Psi(\cdot)$, the first convergence in (35) implies the boundedness of $\{\|g^k_{J_k}\|\}_{\bar{K}}$. Taking into account the boundedness of $\{w^k_{J_k}\}_1^\infty$ (since $0 \le w^k_j \le \delta_k$ for all $j \in J_k$ and all k) we obtain the boundedness of $\{\|v^k_{J_k}\|\}_{\bar{K}}$.

Due to the upper semicontinuity of the generalized Jacobian, the sequence $\{\|H_k^T H_k\|\}$ is bounded for all $k \in \bar{K}$; hence the norm of its submatrix $\{\|(H^k_{.\bar{J}_k})^T H^k_{.\bar{J}_k}\|\}$ is also bounded for $k \in \bar{K}$. The fact

$$\lim_{k\to\infty} \rho(\Psi(w^k)) = \rho(\Psi(w^*))$$

implies that there is a constant $\kappa_1 > 0$ such that

$$\|\mathcal{B}_k\| \le \kappa_1 \tag{36}$$

for all $k \in \bar{K}$. We recall from the proof of Lemma 5.3 that $(g^k)^T v^k = (g^k_{J_k})^T v^k_{J_k} + \|g^k_{\bar{J}_k}\|^2$ and $(g^k_{J_k})^T v^k_{J_k} \ge 0$ for all k. Since w^* is not a stationary point of (3), in view of Lemma 5.2 there exists a constant $\kappa_2 > 0$ such that

$$\max\{(g^k_{J_k})^T v^k_{J_k}, \|g^k_{\bar{J}_k}\|\} \ge \kappa_2 \tag{37}$$

for all $k \in \bar{K}$. By (35) and $\delta_* > 0$, we have $\delta_k \ge \Delta_k$ for all $k \in \bar{K}$ sufficiently large. This implies that the estimate in Proposition 4.3(i) holds for all sufficiently large $k \in \bar{K}$. Hence we have, for all $k \in \bar{K}$ sufficiently large,

$$\begin{aligned} \mathrm{Pred}_k &\ge \Delta_k (g^k_{J_k})^T v^k_{J_k} + \frac{1}{4}\|g^k_{\bar{J}_k}\| \min\left\{\Delta_k, \frac{\|g^k_{\bar{J}_k}\|}{\|\mathcal{B}_k\|}\right\} \ge \frac{1}{4}\gamma_2 \Delta_k, \\ \|\tilde{d}^k\| &\le \min\{1, \Delta_k\}\|v^k_{J_k}\| + \|d^k_{\bar{J}_k}\| \le \left(1 + \|v^k_{J_k}\|\right)\Delta_k, \end{aligned} \tag{38}$$

where the last inequality in (38) uses the bounds (36)–(37) and the limit (35). Then $\{\tilde{d}^k\}_{k\in\bar{K}} \to 0$ because of the boundedness of $\{\|v^k_{J_k}\|\}_{k\in\bar{K}}$ and (35). By the mean value theorem, we have

$$\Psi(w^k + \tilde{d}^k) = \Psi(w^k) + \nabla\Psi(\xi^k)^T \tilde{d}^k \quad \text{for some } \xi^k = w^k + \theta_k \tilde{d}^k,\ \theta_k \in (0,1).$$

Obviously, we have $\{\xi^k\}_{k\in\bar{K}} \to w^*$ as $\{\tilde{d}^k\}_{k\in\bar{K}} \to 0$. Then we obtain for $k \in \bar{K}$ sufficiently large

$$\begin{aligned} |r_k - 1| &= \left|\frac{\Psi(w^k) - \Psi(w^k + \tilde{d}^k)}{\mathrm{Pred}_k} - 1\right| \\ &= \frac{1}{\mathrm{Pred}_k}\left|\Delta_k \left((\nabla\Psi(\xi^k) - \nabla\Psi(w^k))_{J_k}\right)^T v^k_{J_k} + \left((\nabla\Psi(w^k) - \nabla\Psi(\xi^k))_{\bar{J}_k}\right)^T d^k_{\bar{J}_k} \right. \\ &\qquad \left. + \frac{1}{2}(d^k_{\bar{J}_k})^T \mathcal{B}_k d^k_{\bar{J}_k}\right| \qquad (\text{by (35) and the definition of } \tilde{d}^k) \end{aligned}$$

$$\leq \frac{4}{\gamma_2 \Delta_k}\Bigg(\Delta_k \left\|(\nabla\Psi(\xi^k)-\nabla\Psi(w^k))_{J_k}\right\| \|v^k_{\bar J_k}\| + \left\|(\nabla\Psi(w^k)-\nabla\Psi(\xi^k))_{\bar J_k}\right\| \|d^k_{\bar J_k}\| + \frac{1}{2}\|\mathcal{B}_k\|\|d^k_{\bar J_k}\|^2\Bigg) \qquad \text{(by the Cauchy–Schwarz inequality and (38))}$$

$$\leq \frac{4}{\gamma_2}\left((1+\|v^k_{\bar J_k}\|)\|\nabla\Psi(w^k)-\nabla\Psi(\xi^k)\| + \frac{1}{2}\|\mathcal{B}_k\|\Delta_k\|\right) \qquad \text{(by } \Delta_k \geq \|d^k_{\bar J_k}\|)$$

$$\to 0 \qquad \text{(by the boundedness of } \{\|v^k_{\bar J_k}\|\}_{\bar K} \text{ and (35)).}$$

Hence the subsequence $\{r_k\}_{k\in\bar K}$ converges to 1, which is a contradiction to $r_k \leq \rho_1$ in (34). ∎

With the help of Lemma 6.3, we are able to prove the following global convergence result. Its proof is quite standard and is omitted here. One can mimic the proof of [20, Thm. 3.1] to prepare one.

LEMMA 6.4. *Suppose that the whole sequence $\{w^k\}$ was generated according to rule* (19). *Then any accumulation point of $\{w^k\}$ is a stationary point of* (3).

Combining the results of Lemmas 6.1, 6.2, and 6.4, we have our main result in this section.

THEOREM 6.5. *Let $\{w^k\}$ be generated by Algorithm* 5.1, *with the subproblem* (9) *being solved by the truncated CG Algorithm* 4.1. *Then any accumulation point of $\{w^k\}$ is a stationary point of* (3).

7. Local convergence.

Let $\{w^k\}$ be a sequence generated by Algorithm 5.1, and let w^* be a strongly regular solution of (1). Our main result in this section is that if w^* is an accumulation point of $\{w^k\}$, then the whole sequence converges to w^* superlinearly/quadratically. The proof is based on a number of lemmas. The proof techniques of some of those lemmas are borrowed from [19]. Therefore, we will omit most of proofs in this section, but we would like to indicate their connections to [19] and refer to [28] for fully worked out proofs.

The two results of the following lemma are simple consequences of the strong regularity. The first one is about the active set J_* at w^* defined by

$$J_* := \{i \in \mathcal{J} |\ z^*_j = 0\}.$$

Since our algorithm makes use of an active-set strategy, we hope that the set J_k is capable of identifying J_* correctly whenever w^k is close to w^*. This can be shown by using a recently proposed identification technique by Facchinei, Fischer, and Kanzow [10]. The second is the uniform nonsingularity of a matrix sequence [19, Lem. 5].

LEMMA 7.1. *Suppose that $\{w^k\}$ is a sequence generated by Algorithm* 5.1 *and w^* is a strongly regular solution of* (1). *If w^* is an accumulation point of $\{w^k\}$, then the following hold:*

(i) *$J_k = J_*$ for all w^k in a sufficiently small ball around w^*.*

(ii) *There is a constant $c_2 > 0$ such that the matrices $(H^k_{.\bar J_k})^T H^k_{.\bar J_k}$ are nonsingular and*

$$\left\|\left((H^k_{.\bar J_k})^T H^k_{.\bar J_k}\right)^{-1}\right\| \leq c_2$$

for all w^k in a sufficiently small ball around w^*.

Suppose w^* is a solution of $\Phi(w)=0$. Then $\|\Phi(w)\| = o(\sqrt{\|\Phi(w)\|})$ whenever w is close enough to w^*. Using this fact, Proposition 4.3, (6), and Lemma 7.1(ii), we

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

can obtain the following bound on d^k by using a proof technique similar to that of [19, Lem. 6].

LEMMA 7.2. *Suppose that $\{w^k\}$ is a sequence generated by Algorithm* 5.1, *and w^* is a strongly regular solution of* (1). *If w^* is an accumulation point of $\{w^k\}$, then there exists a constant $c_3 > 0$ such that*

$$\|d^k\| \le c_3\sqrt{\|\Phi(w^k)\|}$$

for all w^k sufficiently close to w^*, where d^k denotes the vector computed in step (S.4) *of Algorithm* 5.1.

Using Lemma 7.2, we are now able to show the convergence of the whole sequence $\{w^k\}$ (see [19, Lem. 8] for a proof.)

LEMMA 7.3. *Let $\{w^k\}$ be generated by Algorithm* 5.1, *and let w^* be a strongly regular solution of* (1). *If w^* is an accumulation point of $\{w^k\}$, then the whole sequence $\{w^k\}$ converges to w^*.*

The results developed so far allow us to establish one more technical result, which in turn implies that the iterates are eventually generated by $w^k + d^k$. The third result of the next lemma can be proved similarly to the proof of [19, Lem. 11].

LEMMA 7.4. *Let $\{w^k\}$ be generated by Algorithm* 5.1, *and let w^* be a strongly regular solution of* (1) *and an accumulation point of $\{w^k\}$. Let $\{d^k\}$ denote the directions computed in step* (S.4) *of Algorithm* 5.1. *Then the following hold:*

(i) *For all k sufficiently large, it holds that*

$$\Psi(w^k + d^k) \le \gamma\sqrt{\|\Phi(w^k)\|}.$$

(ii) *There are infinitely many iterates w^k at which* $\mathrm{ind}_k = 0$.

(iii) *It holds that* $\mathrm{ind}_k = 0$ *for all k sufficiently large.*

Proof. Under the assumed conditions, it is proved in Lemma 7.3 that the whole sequence $\{w^k\}$ converges to w^* with $\Phi(w^*) = 0$. Then Lemma 7.2 implies that there exists $c_3 > 0$ such that

$$\|d^k\| \le c_3\sqrt{\|\Phi(w^k)\|}$$

for all k sufficiently large. So the sequence $\{w^k + d^k\}$ also converges to w^*. Then for all k sufficiently large we have

$$\left\{\begin{array}{l} \|\Phi(w^k + d^k)\| \le L\|w^k + d^k - w^*\|, \\ \|\Phi(w^k)\| \ge c_1\|w^k - w^*\|, \ \|H_k\| \le \kappa_3, \\ \sqrt{\|\Phi(w^k)\|} \le \min\{\gamma/(2L^2c_3^2), \ c_1c_3\}, \end{array}\right. \tag{39}$$

where L is the Lipschitz constant of $\Phi(\cdot)$ in a small ball around w^*, c_1 is the constant used in Proposition 2.3, and κ_3 is the constant used in the proof of Lemma 7.2. It also follows from Proposition 2.2 and Lemma 7.1 that for all k sufficiently large

$$\|\Phi(w^k) - \Phi(w^*) - H_k(w^k - w^*)\| = o(\|w^k - w^*\|) \tag{40}$$

and

$$J_k = J_*. \tag{41}$$

(i) The inequalities in (39) yield (i) as follows for all k sufficiently large:

$$\begin{aligned}
&\Psi(w^k + d^k) = \frac{1}{2}\|\Phi(w^k + d^k)\|^2 \\
&\le \frac{L^2}{2}\|w^k + d^k - w^*\|^2 \le \frac{L^2}{2}\left(\|w^k - w^*\| + \|d^k\|\right)^2 \\
&\le \frac{L^2}{2}\left(\|\Phi(w^k)\|/c_1 + c_3\sqrt{\|\Phi(w^k)\|}\right)^2 \le \frac{L^2}{2c_1^2}\|\Phi(w^k)\|\left(c_1c_3 + \sqrt{\|\Phi(w^k)\|}\right)^2 \\
&\le 2(Lc_3)^2\|\Phi(w^k)\| \le \gamma\sqrt{\|\Phi(w^k)\|}.
\end{aligned}$$

(ii) Suppose to the contrary that there is a $\bar{k}$ such that $\mathrm{ind}_k = 1$ for all $k \ge \bar{k}$. We again let K be defined as (22). Then K contains finitely many indices, which we denote by

$$K := \{k_0, k_1, \ldots, k_l\}$$

for some integer l. By the update rule for β_k, we have

$$\beta_k = \beta_{k_l+1} = \Psi(w^{k_l+1}) \quad \forall\, k > k_l + 1,$$

and there is no new update for $\bar{\gamma}$ after $k_l + 1$, i.e.,

$$\bar{\gamma} = \gamma_{k_l+1} = \frac{\Psi(w^{k_l+1})}{\Psi(w^{k_l})} \quad \forall\, k > k_l + 1.$$

Since $\{w^k + d^k\}$ converges to w^*, $\Psi(w^k + d^k)$ converges to $\Psi(w^*) = 0$. Hence for all k sufficiently large

$$\Psi(w^k + d^k) \le \gamma\Psi(w^{k_l}) \le \frac{\gamma}{\bar{\gamma}}\Psi(w^{k_l+1}) = \frac{\gamma}{\bar{\gamma}}\beta_k;$$

that is, test (21) is successful for all k sufficiently large. Since $\mathrm{ind}_k = 1$ for k large enough, Algorithm 5.1 (S.5)(ii) assigns $\mathrm{ind}_{k+1} = 0$, which contradicts our assumption. This establishes (ii). (iii) can be proved similarly to the proof of [19, Lem. 11] by noticing Proposition 4.2. □

We are now at the position to state the main local convergence result.

THEOREM 7.5. *Let $\{w^k\}$ be a sequence generated by Algorithm* 5.1, *and let w^* be a strongly regular solution of* (1) *and an accumulation point of $\{w^k\}$. Then the following statements hold:*

(a) *The whole sequence $\{w^k\}$ converges to w^*.*

(b) *The rate of convergence is Q-superlinear.*

(c) *The rate of convergence is Q-quadratic if, in addition, F is an LC^1 function and h, g are LC^2 functions and $\rho(\Psi(w^k)) = O(\sqrt{\Psi(w^k)})$.*

Proof. Statement (a) follows immediately from Lemma 7.3. We have proved in Lemma 7.4 that $\mathrm{ind}_k = 0$ and test (20) holds for all k sufficiently large. Hence, there exists $\bar{k} > 0$ such that for all $k \ge \bar{k}$

$$w^{k+1} := w^k + d^k.$$

Then by an argument similar to the proof of [19, Lem. 11], we can prove

$$\|w^{k+1} - w^*\| = o(\|w^k - w^*\|),$$

and under the condition in (c) and with Proposition 2.2, we can prove

$$\|w^{k+1} - w^*\| = O(\|w^k - w^*\|^2).$$

This proves (b) and (c). □

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

8. Numerical results. In this section, we present some numerical experiments on a subset of problems from the MCPLIB collection [7]. The details about the implementation are described as follows.

(a) *The penalized Fischer–Burmeister function.* Instead of using the Fischer–Burmeister function φ, we use its penalized version $\varphi_\alpha : \mathbb{R}^2 \to \mathbb{R}$ defined by

$$\varphi_\alpha(a,b) := \alpha\varphi(a,b) + (1-\alpha)a_+b_+, \qquad \alpha \in (0,1],$$

where $a_+ = \max\{0, a\}$ for any $a \in \mathbb{R}$. Numerical tests indicate that the penalized Fischer–Burmeister function usually leads to better numerical performance than the Fischer–Burmeister function [3, 34]. In our implementation, $\alpha = 0.7$, as recommended by Ulbrich [37].

(b) *SSOR preconditioner.* As we pointed out in the introduction, the key issue of efficient implementation of CG-type methods is the *preconditioning.* Steihaug's CG method with preconditioner C can be found in [33]. Although there is no single preconditioning that is "best" for all conceivable types of matrices, we choose C to be the SSOR preconditioner of the following type of linear equations:

$$(A^TA + \rho I)x = b, \tag{42}$$

where A, b have compatible dimension and ρ is a small positive number. This type of equation is exactly what we try to solve at each iteration. Then the SSOR preconditioner corresponds to taking

$$C = P^TP \text{ and } P = D_A^{-1/2}(D_A + \omega L_A^T), \quad 0 \le \omega < 2, \tag{43}$$

with the standard splitting $A^TA + \rho I = L_A + D_A + L_A^T$, where L_A is strictly lower triangular. The cost of matrix-vector product for each of the SSOR PCG iterations including the matrix-vector product with A^TA is in fact only $4\text{nnz}(A)$, and A^TA is not formed explicitly, where nnz is (Matlab) notation of the number of nonzero elements. See [15, Table 1], [1, p. 284], and [2] for more information about the counting. According to [15], SSOR-CG and TMRES (transformed minimal residual algorithm) are the two most efficient methods for solving linear equations of type (42) compared with several other (iterative) methods. Theory and numerical experiments indicate that $\omega = 1$ is often close to the optimum choice of ω [1]. In our implementation, $\omega = 1$.

(c) *Nonmonotone calculation of the reduction-ratio r_k.* In calculating r_k in (18), we used its nonmonotone version,

$$r_k = \left(\mathcal{W}_k - \Psi(w^k + \tilde{d}^k)\right)/\text{Pred}_\text{k},$$

where $\mathcal{W}_k := \max\{\Psi(w^j)|\ j = k+1-\ell, \dots, k\}$ denotes the maximal function value of Ψ over the last ℓ iterations. The nonmonotone version often gives an overall better performance than its monotone version. For more discussion, see [37]. In our implementation, $\ell = 4$.

(d) *Test problems.* The test problems we used are selected from the MCPLIB collection [7] and have at most one bound per variable, i.e., $u_i - l_i = +\infty$ for all i. The collection itself is updated from time to time. As of the initial point of those problems, we follow a suggestion of Ulbrich [37] that interior starting points enable constrained algorithms to identify the correct active constraints more efficiently than starting points close to the boundary. Let $\hat{x}^0$ be the initial point returned by the initialization routine `mcpinit`. Then the initial point chosen is given by $x^0 = \max\{l + 0.1, \min\{u - 0.1, \hat{x}^0\}\}$.

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

The algorithm was implemented in Matlab and run on a SUN Solaris (CDE Version 1.2) workstation. The parameters used are $\Delta_0 = \min\{0.1\|g^0\|, 30\sqrt{10n}\}$, $\Delta_{\min} = 1$, $\rho_1 = 10^{-4}$, $\rho_2 = 0.75$, $\sigma_1 = 0.1$, $\sigma_2 = 10$, $c = 1$, $\gamma = 0.9$, $\delta = 10^{-4}$, and $\text{tol} = 10^{-10}$. The algorithm was terminated if one of the following conditions was met:

$$\max\{\Psi(w^k), \|\nabla\Phi(w^k)\|, \|v_k\|\} \le \text{tol} \;\text{ or }\; \texttt{it\_outer} \ge 100,$$

where `it_outer` denotes the (outer) iteration number. The forcing function used in our implementation is $\rho(\Psi(w)) = \min\{10^{-6}, \sqrt{\Psi(w)}\}$. The stop rule (S.1) used in Algorithm 4.1 is replaced by

$$\|\mathcal{B}_k s^i - g^k_{\bar{J}_k}\|_{C_k} / \|g^k_{\bar{J}_k}\|_{C_k} \le \text{tol}$$

if $g^k_{\bar{J}_k} \neq 0$, and $s = 0$ would otherwise be the (unique) solution of the subproblem.

There are two iterative procedures in the implementation: One is the iterative procedure in Algorithm 5.1, which we call the outer iterative procedure. For each outer iteration, there is the truncated PCG iterative procedure described in Algorithm 4.1 for solving the trust region subproblem, which we call the inner iterative procedure. The average number of inner iterations per outer iteration is essential to the efficiency of our approach. The following data are reported in our numerical results: `n`, the problem size; `it_outer`, number of outer iterations when Algorithm 5.1 was terminated; `it_inner`, average number of PCG iterations per out iteration, i.e., `it_inner` = [Total numbers of PCG iterations/`it_outer`], where $[z]$ denotes the nearest integer to z; `nf`, number of evaluations of the function F; $\Psi(w^f)$, the value of $\Psi(\cdot)$ at the final iterate; $\|\nabla\Psi(x^f)\|$, the value of $\|\nabla\Psi(\cdot)\|$ at the final iterate; $\|v^f\|$, the value of $\|v^k\|$ at the final iteration. `it_outer` is also equal to the number of evaluations of the Jacobian $F'(x)$.

We tested Algorithm 5.1 with two purposes: to demonstrate the importance of preconditioning and to compare our numerical results with existing ones.

(e) *Importance of preconditioning.* For this purpose, we tested three versions of Algorithm 5.1. `tcg`: Algorithm 5.1 without preconditioning ($C = I$); `tcg_ssor`: Algorithm 5.1 with SSOR-preconditioner ($C = P^T P$ with P given by (43)); and `tcg_chol`: Algorithm 5.1 with Cholesky direct factorization ($C = R^T R$ with R being the Cholesky factor of $\mathcal{B}_k$). We expect that `tcg_ssor` is much more efficient than `tcg` and is less efficient than `tcg_chol` (we use `tcg_ssor` as benchmark). The numerical results confirm this expectation. Table 1 contains results from `tcg` and Table 2 contains results from `tcg_ssor`. On the one hand, `tcg` failed to solve four more problems (i.e., `bertsekas`, `freebert`, `games`, and `methan08`) than `tcg_ssor`. But for the remaining solved problems, they behaved quite similarly except for `colvdual`. To solve this problem, `tcg` took many more functional evaluations and outer iterations than `tcg_ssor` did. The observation clearly shows that the preconditioned version is much more efficient than the unpreconditioned version. On the other hand, `tcg_chol` is able to solve one more problem (`ne-hard`) than `tcg_ssor`, and they behaved very similarly for the rest of the problems in terms of number of functional evaluations and outer iterations. For this reason and to save space as well, we did not include the complete results for `tcg_chol`, but the final information for `ne-hard` is included in Table 2. We note that when the Cholesky preconditioner is applied in Algorithm 4.1, in theory it takes only one inner iteration to solve the trust region subproblem, and the resulting direction is of the Gauss–Newton type. Its steplength is controlled

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Table 1
Numerical results with `tcg`.

Problem	n	it_inner	it_outer	nf	$\Psi(x^f)$	$\|\nabla\Psi(x^f)\|$	$\|v^f\|$
badfree	5	2	4	5	2.68e-13	8.87e-07	8.87e-07
bertsekas	15	–	–	–	–	–	–
billups	1	1	1	2	9.80e-05	2.94e-02	0.00e+00
bishopl	1645	–	–	–	–	–	–
colvdual	20	19	48	200	7.11e-13	6.56e-06	6.56e-06
colvnlp	15	18	6	8	3.34e-13	2.91e-05	2.91e-05
cycle	1	1	5	7	2.18e-15	2.44e-07	2.44e-07
degen	2	1	5	7	2.88e-11	7.51e-06	7.51e-06
duopoly	63	–	–	–	–	–	–
ehl_k40	41	–	–	–	–	–	–
ehl_k60	61	–	–	–	–	–	–
ehl_k80	81	–	–	–	–	–	–
ehl_kost	101	–	–	–	–	–	–
explcp	16	8	11	13	1.97e-13	4.40e-07	4.40e-07
forcebsm	184	–	–	–	–	–	–
forcedsa	186	–	–	–	–	–	–
freebert	15	–	–	–	–	–	–
games	16	–	–	–	–	–	–
hanskoop	14	10	22	32	1.65e-12	1.51e-05	1.51e-05
hydroc06	29	94	64	170	8.24e-11	2.59e-04	2.59e-04
hydroc20	99	–	–	–	–	–	–
jel	6	8	6	9	5.84e-14	6.09e-06	6.09e-06
josephy	4	3	4	5	3.38e-20	1.60e-09	1.60e-09
kojshin	4	4	3	4	1.17e-11	2.70e-05	2.70e-05
lincon	419	–	–	–	–	–	–
mathinum	3	3	5	6	8.82e-15	2.63e-07	2.63e-07
mathisum	4	3	8	10	5.83e-11	5.62e-05	5.62e-05
methan08	31	–	–	–	–	–	–
nash	10	10	5	6	1.47e-17	2.87e-07	2.87e-07
ne-hard	3	–	–	–	–	–	–
pgvon106	106	–	–	–	–	–	–
powell	16	12	5	6	6.93e-14	6.14e-06	5.46e-06
powell_mcp	8	7	2	3	3.33e-13	7.37e-06	7.37e-06
qp	4	2	9	45	1.26e-14	8.55e-07	3.19e-07
scarfanum	13	10	13	16	2.83e-16	8.57e-07	8.57e-07
scarfasum	14	12	11	45	7.31e-19	4.28e-08	4.28e-08
scarfbsum	40	–	–	–	–	–	–
shubik	45	–	–	–	–	–	–
simple-ex	17	–	–	–	–	–	–
simple-red	13	15	10	15	2.12e-11	3.10e-06	3.10e-06
sppe	27	36	4	5	1.56e-18	3.58e-09	3.57e-09
tinloi	146	5	6	63	1.30e-11	1.71e-02	1.59e-02
tobin	42	36	8	43	2.02e-22	2.08e-10	1.85e-10
trafelas	2904	–	–	–	–	–	–

by the trust region radius. In practice, it may take more than one inner iteration to produce a direction due to the accumulated roundoff. Moreover, when the linear equations of the type (42) is near singular, the Cholesky factor may not exist, leading to the failure of `tcg_chol`. In our experiments, `tcg_chol` took only one inner iteration per outer iteration. So it is not appropriate to compare `tcg_ssor` with `tcg_chol` in terms of inner iterations taken per outer iteration. However, it is safe to say that the efficiency of the preconditioned Algorithm 5.1 varies with the preconditioners used.

(f) *Comparison.* The results in Table 2 are comparable to those results obtained with existing methods [37, 35]. Moreover, for most of the tested problems the av-

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

TABLE 2
Numerical results with `tcg_ssor`.

Problem	n	it_inner	it_outer	nf	$\Psi(x^f)$	$\|\nabla\Psi(x^f)\|$	$\|v^f\|$
badfree	5	3	4	5	1.15e-12	1.84e-06	1.84e-06
bertsekas	15	7	18	58	8.84e-18	1.43e-07	1.43e-07
billups	1	1	1	2	9.80e-05	2.94e-02	0.00e+00
bishop	1645	–	–	–	–	–	–
colvdual	20	14	22	68	8.05e-13	9.11e-05	9.11e-05
colvnlp	15	11	6	8	3.35e-13	2.92e-05	2.92e-05
cycle	1	1	5	7	2.18e-15	2.44e-07	2.44e-07
degen	2	1	5	7	2.88e-11	7.51e-06	7.51e-06
duopoly	63	–	–	–	–	–	–
ehl_k40	41	–	–	–	–	–	–
ehl_k60	61	–	–	–	–	–	–
ehl_k80	81	–	–	–	–	–	–
ehl_kost	101	–	–	–	–	–	–
explcp	16	4	11	13	1.96e-13	4.39e-07	4.39e-07
forcebsm	184	–	–	–	–	–	–
forcedsa	186	–	–	–	–	–	–
freebert	15	7	24	117	1.18e-12	7.12e-05	7.12e-05
games	16	12	14	36	1.74e-14	5.16e-06	4.58e-06
hanskoop	14	9	24	37	1.09e-13	3.75e-06	3.72e-06
hydroc06	29	41	36	42	6.09e-11	1.86e-07	1.86e-07
hydroc20	99	–	–	–	–	–	–
jel	6	5	6	9	5.84e-14	6.09e-06	6.09e-06
josephy	4	3	4	5	3.38e-20	1.60e-09	1.60e-09
kojshin	4	4	3	4	1.17e-11	2.70e-05	2.70e-05
lincon	419	–	–	–	–	–	–
mathinum	3	3	5	6	8.82e-15	2.63e-07	2.63e-07
mathisum	4	3	8	10	5.83e-11	5.62e-05	5.62e-05
methan08	31	46	51	59	9.08e-11	9.95e-09	9.95e-09
nash	10	4	5	6	1.49e-17	2.86e-07	2.86e-07
ne-hard	3	1	25	51	5.09e-11	5.60e-02	6.60e-02
pgvon106	106	–	–	–	–	–	–
powell	16	10	14	61	5.69e-12	3.80e-05	3.31e-05
powell_mcp	8	7	2	3	3.33e-13	7.36e-06	7.36e-06
qp	4	2	9	45	1.26e-14	8.55e-07	3.19e-07
scarfanum	13	8	13	16	2.83e-16	8.57e-07	8.57e-07
scarfasum	14	9	11	45	7.31e-19	4.28e-08	4.28e-08
scarfbsum	40	–	–	–	–	–	–
shubik	45	–	–	–	–	–	–
simple-ex	17	–	–	–	–	–	–
simple-red	13	10	10	15	1.04e-11	2.18e-06	2.18e-06
sppe	27	15	4	5	1.50e-18	1.96e-09	1.96e-09
tinloi	146	5	6	63	1.30e-11	1.71e-02	1.59e-02
tobin	42	19	8	43	2.78e-23	1.93e-11	1.89e-11
trafelas	2904	–	–	–	–	–	–

erage number of inner iterations per outer iteration (column 3) in Table 2 is small compared with the problem size. Although we have a few more failed problems, we would like to point out that for some of those failed problems, say `ehl`_60, `ehl`_80, and `ehl`_kost, all of which can be solved in [37, 35], we were able to arrive at points with functional values at the order of 10^{-5} and 10^{-8} for `hydroc`20 within 10 iterations, but our algorithm hardly achieved any significant improvement thereafter. The difficulties may come from two resources: On the one hand, there are many ways, all mathematically equivalent, in which to implement the CG method for (42). In *exact* arithmetic they will all generate the same sequence of approximations and all have

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

finite termination property, but in *finite* precision the achieved accuracy may differ substantially. On the other hand, it is hard to select the "best" *preconditioner* for all linear systems of type (42) arising from our implementation. Hence for those failed problems, an appropriate option is an efficient implementation of the CG method with a more suitable *preconditioner* (other than SSOR). Fortunately, many other *preconditioners* with corresponding efficient implementation of the CG method are available; see [1, pp. 293–311]. We also learned that the number of inner iterations may grow significantly for (very) ill-conditioned linear systems of type (42) in order to meet the required accuracy. We refer readers who are interested in the reasons to a recent paper [15] by Hager. We also observed that the sooner direction d^k is taken, the sooner the active set is identified. In fact, for most cases, the active set is identified at least two or three iterations before termination.

Based on the numerical results reported and observation in (e) and (f), we feel that the trust region PCG method proposed in this paper provides an alternative to the existing methods for solving KKT systems. More numerical experiments need to be done to evaluate the proposed approach, especially on large-scale problems with a Jacobian appearing in a certain pattern of sparsity.

9. Conclusions.

In this paper, we proposed a trust region algorithm for solving KKT systems arising from VIPs. Built around those components of the current iterate, which are far from the boundary of the constrained region, the trust region subproblem is solved by the truncated PCG method. Global and local convergence analysis are provided for this method. Numerical experiments show that the proposed method is promising, mainly due to its computational inexpensiveness in that the trust region subproblem is solved by the truncated CG method. We also briefly discussed ways for improving practical efficiency of the proposed method.

Acknowledgments. The authors would like to thank the associate editor and two anonymous referees for their detailed comments, which considerably improved the presentation of the paper. In particular, one referee's expert comments on PCG methods led us to the current version of Algorithm 4.1 as well as condition (13).

REFERENCES

[1] Å. Björck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, PA, 1996.

[2] Å. Björck and T. Elfving, *Accelerated projection methods for computing pseudoinverse solutions of systems of linear equations*, BIT, 19 (1979), pp. 145–163.

[3] B. Chen, X. Chen, and C. Kanzow, *A penalized Fischer-Burmeister NCP-function*, Math. Program., 88 (2000), pp. 211–216.

[4] F.H. Clarke, *Optimization and Nonsmooth Analysis*, Wiley, New York, 1983.

[5] A.R. Conn, N.I.M. Gould, and P.L. Toint, *Global convergence of a class of trust region algorithms for optimization with simple bounds*, SIAM J. Numer. Anal., 25 (1988), pp. 433–460.

[6] A.R. Conn, N.I.M. Gould, and P.L. Toint, *Trust Region Methods*, MPS/SIAM Ser. on Optim. 1, SIAM, Philadelphia, 2000.

[7] S.P. Dirkse and M.C. Ferris, *MCPLIB: A collection of nonlinear mixed complementarity problems*, Optim. Methods Softw., 5 (1995), pp. 319–345.

[8] F. Facchinei, A. Fischer, and C. Kanzow, *A semismooth Newton method for variational inequalities: The case of box constraints*, in Complementarity and Variational Problems: State of the Art, Proc. Appl. Math. 92, M. C. Ferris and J.-S. Pang, eds., SIAM, Philadelphia, 1997, pp. 76–90.

[9] F. Facchinei, A. Fischer, and C. Kanzow, *Regularity properties of a semismooth reformulation of variational inequalities*, SIAM J. Optim., 8 (1998), pp. 850–869.

[10] F. Facchinei, A. Fischer, and C. Kanzow, *On the accurate identification of active constraints*, SIAM J. Optim., 9 (1998), pp. 14–32.

[11] F. Facchinei, A. Fischer, C. Kanzow, and J.-M. Peng, *A simply constrained optimization reformulation of KKT systems arising from variational inequalities*, Appl. Math. Optim., 40 (1999), pp. 19–37

[12] A. Fischer, *A special Newton-type optimization method*, Optimization, 24 (1992), pp. 269–284.

[13] A. Fischer, *Solution of monotone complementarity problems with locally Lipschitzian functions*, Math. Programming, 76 (1997), pp. 513–532.

[14] S.A. Gabriel and J.-S. Pang, *An inexact NE/SQP method for solving the nonlinear complementarity problem*, Comput. Optim. Appl., 1 (1992), pp. 67–91.

[15] W.W. Hager, *Iterative methods for nearly singular linear systems*, SIAM J. Sci. Comput., 22 (2000), pp. 747–766.

[16] H. Jiang, M. Fukushima, L. Qi, and D. Sun, *A trust region method for solving generalized complementarity problems*, SIAM J. Optim. 8 (1998), pp. 140–157.

[17] C. Kanzow, *An inexact QP-based method for nonlinear complementarity problems*, Numer. Math., 80 (1998), pp. 557–577.

[18] C. Kanzow, *Strictly feasible equation-based methods for mixed complementarity problems*, Numer. Math., 89 (2001), pp. 135–160.

[19] C. Kanzow and H.-D. Qi, *A QP-free constrained Newton-type method for variational inequality problems*, Math. Program., 85 (1999), pp. 81–106.

[20] C. Kanzow and M. Zupke, *Inexact trust-region methods for nonlinear complementarity problems*, in Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods, M. Fukushima and L. Qi, eds., Kluwer Academic, Dordrecht, The Netherlands, 1998, pp. 211–233.

[21] C.-J. Lin and J.J. Moré, *Newton's method for large bound-constrained optimization problems*, SIAM J. Optim., 9 (1999), pp. 1100–1127.

[22] J. Liu, *Strong stability in variational inequalities*, SIAM J. Control Optim., 33 (1995), pp. 725–749.

[23] J.J. Moré and D.C. Sorensen, *Computing a trust region step*, SIAM J. Sci. Statist. Comput., 4 (1983), pp. 553–572.

[24] J.-S. Pang and S.A. Gabriel, *NE/SQP: A robust algorithm for the nonlinear complementarity problem*, Math. Programming, 60 (1993), pp. 295–337.

[25] J.-S. Pang and L. Qi, *Nonsmooth equations: Motivation and algorithms*, SIAM J. Optim., 3 (1993), pp. 443–465.

[26] J.-M. Peng, *Global method for monotone variational inequality problems with inequality constraints*, J. Optim. Theory Appl., 95 (1997), pp. 419–430

[27] M.J.D. Powell, *A new algorithm for unconstrained optimization*, in Nonlinear Programming 2, O. L. Mangasarian, R. R. Meyer, and S. M. Robinson, eds., Academic Press, New York, 1975, pp. 1–27.

[28] H.-D. Qi, L. Qi, and D. Sun, *Solving KKT Systems via the Trust Region and the Conjugate Gradient Methods*, AMR99/19, School of Mathematics, University of New South Wales, Sydney, Australia, 1999.

[29] L. Qi, *Convergence analysis of some algorithms for solving nonsmooth equations*, Math. Oper. Res., 18 (1993), pp. 227–244.

[30] L. Qi and H. Jiang, *Semismooth Karush-Kuhn-Tucker equations and convergence analysis of Newton and quasi-Newton methods for solving these equations*, Math. Oper. Res., 22 (1997), pp. 301–325.

[31] L. Qi and J. Sun, *A nonsmooth version of Newton's method*, Math. Programming, 58 (1993), pp. 353–368.

[32] S.M. Robinson, *Strongly regular generalized equations*, Math. Oper. Res., 5 (1980), pp. 43–62.

[33] T. Steihaug, *The conjugate gradient method and trust regions in large scale optimization*, SIAM J. Numer. Anal., 20 (1983), pp. 626–637.

[34] D. Sun and L. Qi, *On NCP-functions*, Comput. Optim. Appl., 13 (1999), pp. 201–220.

[35] D. Sun, R.S. Womersley, and H.-D. Qi, *A feasible semismooth asymptotically Newton method for mixed complementarity problems*, Math. Program., 94 (2002), pp. 167–187.

[36] P.L. Toint, *Towards an efficient sparsity exploiting Newton method for minimization*, in Sparse Matrices and Their Uses, I. Duff, ed., Academic Press, New York, 1981, pp. 57–88.

[37] M. Ulbrich, *Nonmonotone trust-region methods for bound-constrained semismooth equations with applications to nonlinear mixed complementarity problems*, SIAM J. Optim., 11 (2001), pp. 889–917.

[38] Y. Yuan, *On the truncated conjugate gradient method*, Math. Program., 87 (2000), pp. 561–573.

Downloaded 10/29/12 to 158.132.161.52. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php