数理解析研究所講究録
1079 巻 1999 年 1-14

A hybrid method for solving a general constrained optimization problem [1]

November 16, 1998

Nobuko SAGARA (相良信子)

Department of Management, Aichi University

Masao FUKUSHIMA (福嶋雅夫)

Department of Applied Mathematics and Physics, Kyoto University

1. Introduction

Recently we have proposed an algorithm for solving nonlinear least squares problems with linear constraints [15]. The method proposed there successively constructs trust region constraints, which are ellipsoids centered at the iterative points, in such a way that they lie in the relative interior of the feasible region. The purpose of this paper is to generalize the results of [15] to nonlinear equality constrained problems with non-negative variables.

Over the last decade, various methods using a trust region strategy have been studied for constrained nonlinear optimization problems, including box constrained problems [5, 6, 11, 14, 16] and nonlinear equality constrained ploblems [2, 4, 7, 8, 9, 13, 17, 19]. In particular, when dealing with equality constraints, adding a trust region constraint directly to the problem may result in an infeasible subproblem. To overcome this difficulty, two approaches have been introduced. The first approach, which was first proposed by Vardi [17], and later used by Byrd, Schnabel and Shultz [2] and M. El-Alem [7], relaxes the linearized equality constraints so as to intersect a trust region constraint. Specifically we first compute the vertical component of a trial step in the range space of the matrix of equality constraint gradients at each iteration and then the horizontal (tangential) component in the null space. The second approach, which was introduced by Celis, Dennis and

[1]This work was supported in part by the Scientific Grant-in-Aid C-59 from Aichi University.

Tapia [4] and later studied by Powell and Yuan [13], iteratively solves quadratic programming subproblems with some additional constraints using a standard trust region strategy. We adopt the second approach in this paper.

In this paper, we consider the following nonlinearly constrained optimization problem:

$$\text{minimize } f(x) \quad \text{subject to} \quad c(x) = 0, \quad x \geq 0, \tag{1.1}$$

where $x \in R^n$ and $c(x) = (c_1(x), c_2(x), ..., c_m(x))^T$. We assume that $f : R^n \to R$ and $c_j : R^n \to R$, $j = 1, ..., m$, are continuously differentiable. Note that any optimization problem involving inequality constraints can be transformed into the form (1.1) by using slack variables.

The main aim of this paper is to extend the methods presented in [14, 15] to the general constrained optimization problem (1.1). Similar to the methods of [14, 15], the proposed method constructs trust region constraints that are ellipsoids centered at the iterative points, in such a way that they lie in the interior of the non-negative orthant. Thus the method belongs to the class of interior point methods. However, since solutions of the problem are usually on the boundary of the non-negative orthant, the trust region ellipses may tend to be arbitrarily thin and, as a result, we eventually suffer from numerical instability. To avoid this difficulty, we incorporate the idea of active set strategy into the method. With such a modification, we may still expect that the method retains the advantage of the interior point method, as observed in our previous papers [14, 15] for problems with simple bound constraints on the variables, and problems with linear inequality constraints and non-negative variables.

The paper is organized as follows. Section 2 describes the algorithm. In Section 3, we report some computational results.

2. Description of the Algorithm

Suppose that x is a current point such that $x > 0$. Let p denote the vector which determines the next point x^+ from the current point x, that is, $x^+ = x + p$.

We consider the following subproblem involving a trust region constraint:

$$\text{minimize}_p \qquad g(x)^T p + \tfrac{1}{2} p^T W p \tag{2.1a}$$

$$\text{subject to} \quad \| c(x) + A(x)^T p \| \leq \theta, \tag{2.1b}$$

$$\| D(x)p \| \leq \Delta, \tag{2.1c}$$

where $g(x) = \nabla f(x)$, $A(x) = \nabla c(x)$, $D(x) = diag(1/x_i)$, θ and Δ are parameters such that $\theta > 0$ and $0 < \Delta < 1$, and W is an $n \times n$ symmetric matrix. Note that θ, Δ and W are dependent on the iteration. In particular, θ is supposed to satisfy the inequalities

$$\min_{\|D(x)p\| \leq b_1 \Delta} \| c(x) + A(x)^T p \| \leq \theta \leq \min_{\|D(x)p\| \leq b_2 \Delta} \| c(x) + A(x)^T p \|, \tag{2.2}$$

where b_1 and b_2 are given constants such that $0 \leq b_2 \leq b_1 \leq 1$. Note that if p satisfies (2.1c) then $x^+ := x + p$ remains in the positive orthant, i.e., $x^+ > 0$ (see [1]).

The role of the constraint (2.1b) may be explained as follows. When using the trust region constraint $\|D(x)p\| \leq \Delta$, the linearized constraints $c(x) + A(x)^T p = 0$ may have no solution within the trust region. Attempts to overcome this difficulty have been made by several authors [13, 17, 2, 4]. In particular, introducing the parameter θ satisfying (2.2) was proposed by Powell and Yuan [13]. Note that, if we assume $b_1 = 1$ and $b_2 = 0$, then (2.2) reduces to

$$\min_{\|D(x)p\| \leq \Delta} \| c(x) + A(x)^T p \| \leq \theta \leq \| c(x) \|,$$

which ensures the existence of a feasible solution to (2.1)

The trust region constraint $\|D(x)p\| \leq \Delta$ in (2.1) represents an ellipsoid, which is centered at the current point x and strictly contained in the interior of the non-negative orthant $\{x \in R^n | x \geq 0\}$. However, when the current point is close to the boundary of the non-negative orthant, the trust region ellipsoid becomes thin and solution of (2.1) suffers from numerical instability. To overcome this dificulty, we modify (2.1) using the idea of active set strategy [10] in constrained optimization.

For a current point $x > 0$, we define the set of indices

$$I = \{i | x_i \geq \epsilon_1\}, \tag{2.3}$$

where ϵ_1 is a sufficiently small positive constant. We also denote $\bar{I} = \{1, 2, ..., n\} - I$. According to the above definition, we partition the vectors $x, p, g(x)$ and the matrices $W, A(x)$ as

$$x = \begin{bmatrix} x_I \\ x_{\bar{I}} \end{bmatrix}, \quad p = \begin{bmatrix} p_I \\ p_{\bar{I}} \end{bmatrix}, \quad g(x) = \begin{bmatrix} g_I(x) \\ g_{\bar{I}}(x) \end{bmatrix}, \quad W = \begin{bmatrix} W_{II} & W_{I\bar{I}} \\ W_{\bar{I}I} & W_{\bar{I}\bar{I}} \end{bmatrix}, \quad A(x) = \begin{bmatrix} A_I(x) \\ A_{\bar{I}}(x) \end{bmatrix}.$$

By adding the extra constraints $p_{\bar{I}} = 0$ to (2.1), we get the following problem:

$$\begin{aligned} \text{minimize}_{p_I} \quad & g_I(x)^T p_I + \frac{1}{2} p_I^T W_{II} p_I \\ \text{subject to} \quad & \| \, c(x) + A_I(x)^T p_I \, \| \leq \theta, \\ & \| \, D_I(x) p_I \, \| \leq \Delta, \end{aligned} \tag{2.4}$$

where θ is chosen, similar to (2.2), to satisfy the condition

$$\min_{\|D_I(x)p_I\| \leq b_1 \Delta} \| \, c(x) + A_I(x)^T p_I \, \| \leq \theta \leq \min_{\|D_I(x)p_I\| \leq b_2 \Delta} \| \, c(x) + A_I(x)^T p_I \, \|, \tag{2.5}$$

and b_1, b_2 and Δ are parameters such that $0 \leq b_2 \leq b_1 \leq 1$ and $0 < \Delta < 1$, and $D_I(x)$ is the diagonal submatrix of $D(x)$ with elements $1/x_i, \ i \in I$.

Let p_I be a solution of (2.4). To test whether we should accept the trial step $p = (p_I, 0)^T$, we use Fletcher's differentiable exact penalty function

$$\phi\,(x; \sigma) = f(x) - \lambda(x)^T c(x) + \sigma \|c(x)\|^2, \tag{2.6}$$

where $\sigma > 0$ is a penalty parameter and $\lambda(x)$ is an estimate of the vector of Lagrange multipliers defined by

$$\lambda(x) = \left(A_I(x)^T A_I(x) \right)^+ A_I(x)^T g_I(x), \tag{2.7}$$

where A^+ is the generalized inverse of the matrix A. The vector $\lambda(x)$ minimizes the sum of squared residuals of the Kuhn-Tucker conditions for (2.4), that is,

$$\lambda(x) = \text{argmin}_{\lambda \in R^m} \|g_I(x) - A_I(x)\lambda\|^2.$$

We define the predicted reduction of $\phi(x + p; \sigma) - \phi(x; \sigma)$ by

$$\begin{aligned} \psi_I(x,p) \;\; = \;\; & (g_I(x) - A_I(x)\lambda(x))^T p_I + \frac{1}{2} p_I^T W_{II} \hat{p}_I \\ & - [\lambda(x+p) - \lambda(x)]^T \left[c(x) + \frac{1}{2} A_I(x)^T p_I \right] \\ & + \sigma \left(\|c(x) + A_I(x)^T p_I\|^2 - \|c(x)\|^2 \right), \end{aligned} \tag{2.8}$$

where $\hat{p}_I$ is the orthogonal projection of p_I onto the null space of $A_I(x)^T$. Note that a similar formula for the predicted reduction has been used in [13] and [7]. Here the parameter σ needs to be chosen so that $\psi_I(x,p) < 0$. More specifically, if the inequality

$$\psi_I(x,p) \leq \frac{1}{2}\sigma\left(\|c(x) + A_I(x)^T p_I\|^2 - \|c(x)\|^2\right) \tag{2.9}$$

is satisfied, then σ remains the same. Otherwise, increase σ to the value

$$\sigma := 2\sigma + \max\left\{0, \frac{2\psi_I(x,p)}{\|c(x)\|^2 - \|c(x) + A_I(x)^T p_I\|^2}\right\}, \tag{2.10}$$

which ensures that $\psi_I(x,p)$ with the updated σ satisfies (2.9). Note that the right-hand side of (2.9) is negative whenever $p_I \neq 0$. This can be shown in a way similar to Lemma 3.3 in [13].

Next we compute the ratio

$$\rho \equiv \frac{\phi\,(x+p;\sigma) - \phi\,(x;\sigma)}{\psi_I(x,p)}. \tag{2.11}$$

If $\phi(x+p;\sigma)$ is sufficiently smaller than $\phi(x;\sigma)$, then we accept p to determine the next point. Otherwise, we halve Δ and solve subproblem (2.4) again. More precisely, let $0 < \mu < \eta < 1$, $\gamma > 1$ and $0 < \Delta_{max} < 1$ be given constants. Compute the ratio ρ defined by (2.11). If $\rho < \mu$, then put $x^+ := x$ and $\Delta^+ := 1/2\Delta$, and then solve subproblem (2.4); if $\mu \leq \rho < \eta$, then put $x^+ := x + p$ and $\Delta^+ := \Delta$; if $\rho \geq \eta$, then put $x^+ := x + p$ and $\Delta^+ := \min(\gamma\Delta, \Delta_{max})$. To test whether the new point $x := x^+$ is an approximate optimal solution associated with the current index set I, we check the inequality

$$\|c(x)\| + \|g_I(x) - A_I(x)\lambda(x)\| < \epsilon_2, \tag{2.12}$$

where $\epsilon_2 > 0$ is a predetermined positive number significantly smaller than ϵ_1. If (2.12) is not satisfied, we update the index set I from the new iterative point x. As a result, we have subproblem (2.4) corresponding to the new index set I. In this manner, we repeatedly solve subproblems (2.4) as long as (2.12) does not hold. On the other hand, if (2.12) is satisfied, then we proceed to examine optimality of the point x for the original problem (1.1). Namely, we compute an estimate of Lagrange multipliers $\lambda(x)$ by (2.7) and check if $\lambda(x)$ satisfies the inequality

$$g_{\bar{I}}(x) - A_{\bar{I}}(x)\lambda(x) > -\epsilon_3 e, \tag{2.13}$$

where ϵ_3 is a sufficiently small positive constant and e is a vector of appropriate dimension whose components are all unity. When the condition (2.13) is violated, we choose an index i from $\bar{I}$ such that

$$i := \operatorname{argmax} \left\{ \left| (g_{\bar{I}}(x) - A_{\bar{I}}(x)\lambda(x))_i \right| \;\middle|\; (g_{\bar{I}}(x) - A_{\bar{I}}(x)\lambda(x))_i \le -\epsilon_3,\ i \in \bar{I} \right\}, \tag{2.14}$$

add it to the current inactive index set I, namely,

$$I := I \cup \{i\}, \qquad \bar{I} := \bar{I} - \{i\},$$

and try to improve the current solution by solving subproblem (2.4) with the updated index set I.

If the point x satisfies both conditions (2.12) and (2.13), then we call it an ϵ-optimal solution for problem (1.1), with $\epsilon = (\epsilon_1, \epsilon_2, \epsilon_3)$.

We are now ready to state an algorithm for finding an ϵ-optimal solution of (1.1).

Algorithm

Initialize: Choose sufficiently small positive constants ϵ_i, $i = 1, 2, 3$ and parameters μ, η, γ and Δ_{max} such that $0 < \mu < \eta < 1, \gamma > 1$ and $0 < \Delta_{max} < 1$. Also, choose parameters b_1 and b_2 with $0 < b_2 \le b_1 < 1$ to get θ in subproblem (2.4). Let $x \ge \epsilon_1 e$, $\Delta \in (0, 1)$, and $\sigma > 0$ be an initial feasible interior point, an initial trust region radius, and an initial penalty parameter, respectively. Let I be the index set defined by (2.3) and $\bar{I}$ be the complement of I.

while x is not ϵ-optimal **do** /outer **while** loop/

begin

1 **while** $\|c(x)\| + \|g_I(x) - A_I(x)\lambda(x)\| \ge \epsilon_2$ **do** /inner **while** loop/

begin

2 Solve subproblem (2.4) to find p_I ;

3 $p_{\bar{I}} := 0$;

4 Compute $\psi_I(x, p)$ by (2.8)

5 **if** (2.9) does not hold then

begin

6 Update σ by (2.10);

end

endif

7 Compute ρ by (2.11)

8 **if** $\rho \geq \mu$ **then**
begin
9 $x := x + p$
10 Update matrix W
11 **if** $\rho \geq \eta$ **then**
12 $\Delta := \min(\gamma\Delta, \Delta_{max})$
endif
end
13 $I := I - \{i \in I | x_i < \epsilon_1\}$
else
14 $\Delta := \frac{1}{2}\Delta$
go to 2
endif
end
endwhile /end inner **while** loop/
15 **if** (2.13) is violated **then**
(Comment: if (2.13) holds, the whole algorithm is terminated with an ϵ-optimal solution.)
begin
16 $i := \arg\max \left\{ \left| (g_{\bar{I}}(x) - A_{\bar{I}}(x)\lambda(x))_i \right| \;\middle|\; (g_{\bar{I}}(x) - A_{\bar{I}}(x)\lambda(x))_i < -\epsilon_3,\ i \in \bar{I} \right\}$;
17 $I := I \cup \{i\}$
end
endif
end

endwhile /end outer **while** loop/

Note that the sequence of iterates x generated by this algorithm lies in the positive orthant. Here, the inner loop tries to find the solution of the equality constrained optimization problem with some index set I.

3. Numerical Experiments

We executed numerical experiments with the algorithm proposed in Section 2. In this algorithm, the most time consuming task is to solve subproblem (2.4) at each iteration, which is the problem of minimizing a convex quadratic function subject to two quadratic constraints. To solve (2.4) we used the approach by Zhang [18] to reformulate the problem into a univariate nonlinear equation which is continuous, at least piecewise differentiable and monotone.

The program of the algorithm was coded in Fortran77. The computation was carried out using double precision arithmetic on a FACOM-M382 Computer at the Data Processing Center of Kyoto University.

In order to examine performance of the proposed algorithm, we have solved the following four test problems. These test problems except Example 3 have been constructed by modifying the problems in [3]. In particular, the last problem is an expansion of the large scale problem No.7 in [3]. Throughout the experiments, we set the parameter values as follows: $\epsilon_1 = 0.001$, $\epsilon_2 = 0.01$, $\epsilon_3 = 0.001$, $\mu = 0.2$, $\eta = 0.6$, $\gamma = 1.2$ and $\Delta_{max} = 0.99$. Also we set $b_1 = 0.8$ and $b_2 = 0.2$, and let θ be the mean value of the both sides of (2.5). The symmetric matrix W_{II} in subproblem (2.4) was updated by BFGS formula.

Example 1.

$$
\begin{aligned}
\text{minimize } f(x) &= (x_1-1)^2+(x_1-x_2)^2+(x_2-x_3)^2+(x_3-x_4)^4+(x_4-x_5)^4 \\
&\quad +(x_5-x_6)^4+(x_6-x_7)^4+(x_7-x_8)^4+(x_8-x_9)^4+(x_9-x_{10})^4 \\
\text{subject to } c_1(x) &= x_1+x_2^2+x_3^3-2-\sqrt{18}=0 \\
c_2(x) &= x_2-x_3^2+x_4+2-\sqrt{8}=0 \\
c_3(x) &= x_1x_5-2=0 \\
c_4(x) &= x_2x_6-3=0 \\
c_5(x) &= x_3x_7-4=0 \\
c_6(x) &= x_4x_8-5=0 \\
c_7(x) &= x_5x_9-6=0
\end{aligned}
$$

$$
\begin{aligned}
c_8(x) &= x_6 x_{10} - 7 = 0 \\
x_i &\geq 0, \ \ i = 1, 2, ..., 10.
\end{aligned}
$$

The numerical results for several initial points x^0 are shown given in the following table.

x^0	iter.	CPU time
(1,...,1)	11	92
(2,...,2)	9	54
(3,...,3)	13	70
(4,...,4)	13	86
(5,...,5)	17	104
(6,...,6)	18	100
(7,...,7)	19	99
(8,...,8)	20	116

iter. = number of iterations

CPU time = CPU time in msec

Example 2.

$$
\begin{aligned}
\text{minimize } f(x) &= x_1 x_2 x_3 + x_3 x_4 x_5 + x_5 x_6 x_7 + \cdots + x_{n-2} x_{n-1} x_n \\
\text{subject to } c_1(x) &= x_1 x_n - 1 = 0 \\
c_2(x) &= x_2 x_{n-1} - 2 = 0 \\
c_3(x) &= x_3 x_{n-2} - 3 = 0 \\
&\vdots \\
c_{m-1}(x) &= x_{m-1} x_{m+1} - (m-1) = 0 \\
c_m(x) &= x_m^2 - m = 0 \\
x_i &\geq 0, \ \ i = 1, 2, ..., n,
\end{aligned}
$$

where n is odd, $m = [\frac{n}{2}]+1$ and $[k]$ is the maximum integer not exceeding k. We experimented with an initial vector $x^0 \in R^n$ whose elements are all 2.0. The numerical results are shown in the following table.

n	m	iter.	CPU time
9	5	5	17
11	6	5	21
13	7	5	27
15	8	6	43
17	9	6	60
19	10	7	98
21	11	7	152
23	12	8	241
25	13	8	339
29	15	9	675
31	16	7	414

iter. = number of iterations

CPU time = CPU time in msec

Example 3.

$$\begin{aligned}
\text{minimize } f(x) &= (x_1-1)^2+(x_1-x_2)^2+\cdots+(x_{n-1}-x_n)^2 \\
\text{subject to } c_1(x) &= x_1+x_2+\cdots+x_{n-1}-5n=0 \\
c_2(x) &= x_1^2+x_2^2+\cdots+x_{\frac{n}{2}-2}-20n=0 \\
c_3(x) &= x_{\frac{n}{2}+3}^2+\cdots+x_n^2-20n=0 \\
x_i &\geq 0, \;\; i=1,2,...,n,
\end{aligned}$$

where n is even. We experimented with an initial vector $x^0 \in R^n$ whose elements are all 5.0. The numerical results are shown in the following table.

n	iter.	CPU time	$\|I\|$
10	77	243	8
20	80	969	14
30	54	1898	21
40	78	6365	27
50	75	10424	34
60	82	18882	40
70	89	33071	46
80	87	39488	53
85	92	57826	56
90	99	57907	59

iter. = number of iterations

CPU time = CPU time in msec

$|I|$ = number of indices i such that $x_i \geq \epsilon_1$.

Example 4.

$$\begin{aligned}
\text{minimize } f(x) &= \frac{100}{2}\sum_{j=1}^{L}(x_{5j+2} - x_{5j+1}^2)^2 + \frac{1}{2}\sum_{j=1}^{L}(1 - x_{5j+1})^2 \\
\text{subject to } c_{3k-2}(x) &= x_{5k-4}x_{5k-3} - 1.0 + x_{5k-2}^2 = 0 \\
c_{3k-1}(x) &= x_{5k-3}^2 + x_{5k-4} - x_{5k-1}^2 = 0 \\
c_{3k}(x) &= -x_{5k-4} - x_{5k}^2 + 0.5 = 0, \quad k = 1, 2, ..., L \\
x_i &\geq 0, \quad i = 1, 2, ..., n,
\end{aligned}$$

where L is a positive integer, $n = 5L$ and $m = 3L$.

The numerical results are summarized in the following table.

L	n	m	x^0	iter.	CPU time	B/A
5	25	15	(0.5,...,0.5)	34	1055	
			(0.6,...,0.6)	34	1048	0.0019
			(1.0,...,1.0)	37	1256	
			(1.5,...,1.5)	37	1415	
			(2.0,...,2.0)	38	1239	
10	50	30	(0.5,...,0.5)	44	8547	
			(0.6,...,0.6)	44	7813	0.0008
			(1.0,...,1.0)	47	9102	
			(1.5,...,1.5)	49	9942	
			(1.9,...,1.9)	51	11442	
20	100	60	(0.5,...,0.5)	57	80684	
			(0.6,...,0.6)	56	82920	0.0003
			(0.9,...,0.9)	58	90307	
50	250	150	(0.5,...,0.5)	77	1834208	
			(0.6,...,0.6)	76	1822075	0.0001
			(0.8,...,0.8)	77	1892914	
60	300	180	(0.6,...,0.6)	81	3374823	0.0001

iter. = number of iterations

CPU time = CPU time in msec to get an ϵ-solution

A = CPU time to get an ϵ-solution

B = CPU time to recover an accurate solution from an ϵ-solution

For this example, we considered a correction phase that recovers more accurate solution from the obtained ϵ-optimal solution, because the proposed method merely finds ϵ-optimal solutions. The CPU time spent (B) in the correction phase was always less 1% of the CPU time (A) spent to obtain an ϵ-solution. Moreover the ratio B/A decreases as the problem size increases. Over all, the proposed method solved successfully the above test problems even if a starting point was far from the solution or the problem size were large, though it consumed relatively large computation time.

References

[1] E. R. Barnes, A Variation on Karmarkar's Algorithm for Solving Linear Programming Problem. *Mathematical Programming*, Vol. 36 (1986), 174-182.

[2] R. H. Byrd, R. B. Schnabel and G. A. Shultz, A Trust Region Algorithm for Nonlinearly Constrained Optimization. *SIAM Journal on Numerical Analysis*, Vol. 24 (1987), 1152-1170.

[3] P. T. Boggs and J. W. Tolle, A Strategy for Global Convergence in Sequential Quadratic Programming Algorithm. *SIAM Journal on Numerical Analysis*, Vol.26 (1989), 600-623.

[4] M. R. Celis, J. E. Dennis and R. A. Tapia, A Trust Region Strategy for Nonlinear Equality Constrained Optimization. *Numerical Optimization* (eds. P. T. Boggs, R. H. Byrd and R. B. Schnabel), SIAM, Philadelphia, 1985, 71-82

[5] A. R. Conn, N. I. M. Gould and Ph. L. Toint, Global Convergence of a Class of Trust Region Algorithms for Optimization with Simple Bounds. *SIAM Journal on Numerical Analysis*, Vol. 25 (1988), 433-460.

[6] A. R. Conn, N. I. M. Gould and Ph. L. Toint, Testing a Class of Methods for Solving Minimization Problems with Simple Bounds on the Variables. *Mathematics of Computation*, Vol. 50 (1988), 399-430.

[7] M. El-Alem, A Robust Trust-Region Algorithm with Nonmonotonic Penalty Parameter Scheme for Constrained Optimization. *SIAM Journal on Optimization*, Vol. 5 (1995), 348-378.

[8] M. El-Alem, A Global Convergence Theory for the Celis-Dennis-Tapia Trust-Region Algorithm for Constrained Optimization. *SIAM Journal on Numerical Analysis*, Vol. 28 (1991), 266-290.

[9] M. M. El-Alem and R. A. Tapia, Numerical Experience with a Polyhedral-Norm CDT Trust-Region Algorithm. *Journal of Optimizatin Theory and Applications*, Vol. 85 (1995), 575-591.

[10] P. E. Gill, W. Murray and M. H. Wright, *Practical Optimization.* Academic Press, London, 1981.

[11] M. Lescrenier, Convergence of Trust Region Algorithms for Optimization with Bounds with Strict Complementarity Does Not Hold. *SIAM Journal on Numerical Analysis*, Vol. 28 (1991), 476-495.

[12] J. M. Martínez and S. A. Santos, A Trust-Region Strategy for Minimization on Arbitrary Domains. *Mathematical Programming*, Vol. 68 (1995), 267-301.

[13] M. J. D. Powell and Y. Yuan, A Trust Region Algorithm for Equality Constrained Otimization. *Mathematical Programming*, Vol. 49 (1991), 182-211.

[14] N. Sagara and M. Fukushima, A Hybrid Method for the Nonlinear Least Squares Problem with Simple Bounds. *Journal of Computational and Applied Mathematics*, Vol. 36 (1991), 149-157.

[15] N. Sagara and M. Fukushima, A Hybrid Method for the Nonlinear Least Squares Problem with Linear Inequality Constraints. *Journal of the Operations Research Society of Japan*, Vol. 38 (1995), 55-69.

[16] Ph. L. Toint, Global Convergence of a Class of Trust Region Methods for Nonconvex Minimization in Hilbert Space. *IMA Journal of Numerical Analysis*, Vol. 8 (1988), 231-252.

[17] A. Vardi, A Trust Region Algorithm for Equality Constrained Minimization: Convergence Properties and Implementation. *SIAM Journal on Numerical Analysis*, Vol. 22 (1985), 575-591.

[18] Y. Zhang, Computing a Celis-Dennis-Tapia Trust-Region Step for Equality Constrained Optimization. *Mathematical Programming*, Vol. 55 (1992), 109-124.

[19] J. Z. Zhang and D. T. Zhu, Projected Quasi-Newton Algorithm with Trust Region for Constrained Optimization. *Journal of Optimizatin Theory and Applications*, Vol. 67 (1990), 369-393.