Journal of Computational Mathematics, Vol.15, No.1, 1997, 36–54.

QP-FREE, TRUNCATED HYBRID METHODS FOR LARGE-SCALE NONLINEAR CONSTRAINED OPTIMIZATION*[1)]

Q. Ni
(*School of Science, Nanjing University of Aeronautics and Astronautics, Nanjing, China*)

Abstract

In this paper, a truncated hybrid method is proposed and developed for solving sparse large-scale nonlinear programming problems. In the hybrid method, a symmetric system of linear equations, instead of the usual quadratic programming subproblems, is solved at iterative process. In order to ensure the global convergence, a method of multiplier is inserted in iterative process. A truncated solution is determined for the system of linear equations and the unconstrained subproblems are solved by the limited memory BFGS algorithm such that the hybrid algorithm is suitable to the large-scale problems. The local convergence of the hybrid algorithm is proved and some numerical tests for medium-sized truss problem are given.

1. Introduction

In this paper we consider the following nonlinear programming problem

$$\begin{aligned} &\text{minimize} && f(x) \\ &\text{subject to} && g_j(x) \geq 0, \quad j \in J = \{1, \ldots, m\}. \end{aligned} \tag{1.1}$$

Extensions to problem including also equality constraints will be possible. The function $f : R^n \to R^1$ and $g_j : R^n \to R^1,\ j \in J$ are twice continuously differentiable. In particular, we apply QP-free (without quadratic programming subproblems), truncated hybrid methods for solving the large-scale nonlinear programming problems, in which the number of variables and the number of constraints in (1.1) are great. We discuss the case, where second derivatives in (1.1) are sparse and easy to be obtained.

Many iteration methods for solving (1.1) needs to solve quadratic programming (QP) subproblems at each iteration (see [**?**], [**?**], [**?**], [**?**], [**?**]). For large-scale case it is relative expensive. In a class of hybrid methods, proposed and developed in [**?**], [**?**], [**?**], [**?**] and [**?**], the subproblem is replaced with a symmetric system of not more than

* Received October 29, 1994.

1) The research was supported by the State Education Commission Grant for returned scholars and was carried out while the author was at the State Key Laboratory of Scientific and Engineering Computing, ICMSEC, Chinese Academy of Sciences, Beijing, China.

$n + m$ linear equations. In order to apply the class of hybrid methods to large-scale constrained optimization, we consider the following modifications.

(1) Instead of an exact solution, a truncated solution is determined for a system of linear equations, which is regarded as a subproblem of (1.1). This is because computing an exact solution by using a direct method such as Gaussian elimination can be expensive for large-scale problem. We research the termination criteria of the subproblem and the tradeoff between the amount of work required to compute a update direction and the accuracy with which the subproblem is solved.

(2) The conjugated gradient method is chosen as a iterative method for solving the system of linear equations.

(3) We choose the method of multiplier as a globally convergent method in hybrid method. Because the ill-conditioning can be avoided, the unconstrained subproblems are easily solved by a limited memory quasi-Newton method.

(4) In order to guarantee the numerical stability, the index set is modified and some approximated Lagrangian multipliers at each iteration are corrected such that we avoided this case where the denominators in the numerical computation is too small.

In addition, the convergence rate is proved in detail.

For the following investigation we require some notations and assumptions. The Lagrangian of problem (1.1) is defined by

$$L(x,u) = f(x) - \sum_{j=1}^{m} u_j g_j(x),$$

and its first-order and second-order derivatives with respect to the first argument are denoted by

$$\begin{aligned} \bigtriangledown_x L(x,u) &= \bigtriangledown f(x) - \sum_{j=1}^{m} u_j \bigtriangledown g_j(x), \\ \bigtriangledown^2_{xx} L(x,u) &= \bigtriangledown^2 f(x) - \sum_{j=1}^{m} u_j \bigtriangledown^2 g_j(x), \end{aligned}$$

where $u \in R^m$ is an approximation of Lagrangian multiplier vector of (1.1). With this notation, a pair (x^*, u^*) is called a Kuhn-Tucker pair of (1.1) if the following Kuhn-Tucker conditions hold:

$$\begin{aligned} \bigtriangledown_x L(x^*,u^*) = 0, \quad u_j^* g_j(x^*) = 0, \quad j \in J, \\ u_j^* \geq 0, \quad g_j(x^*) \geq 0, \quad j \in J. \end{aligned}$$

The Kuhn-Tucker conditions are equivalent to the system

$$\begin{aligned} 0 = \qquad P(x,u,t) = \begin{bmatrix} \bigtriangledown_x L(x,u) \\ - - - - - - - - - \\ t_j - g_j(x) \\ - - - - - - - - - \\ u_j t_j \end{bmatrix} \\ u_j \geq 0, \quad t_j \geq 0, \quad j \in J, \end{aligned} \tag{1.2}$$

where $P : R^{n+2m} \to R^{n+2m}$.

The following assumptions are satisfied at a Kuhn-Tucker pair (x^*, u^*).

Assumption 1 (A1)

The functions $\bigtriangledown^2 f$ and $\bigtriangledown^2 g_j$ $(j \in J)$ are Lipschitz-continuous in a neighbourhood of x^*, i.e. there are positive number R and L such that

$$\max\{\| \bigtriangledown^2 f(x) - \bigtriangledown^2 f(\bar{x})\|,\ \| \bigtriangledown^2 g_j(x) - \bigtriangledown^2 g_j(\bar{x})\|,\ j \in J\} \leq L\|x - \bar{x}\|$$

for all $x,\ \bar{x} \in B(x^*, R)$. The set $B(x^*, R) = \{x :\ \|x - x^*\| \leq R\}$ denotes the closed ball with the center x^* and the radius R. For vector the Euclidean norm is chosen, the used matrix norm is assumed to be compatible.

Assumption 2 (A2)

The Kuhn-Tucker pair (x^*, u^*) satisfied the Jacobian uniqueness conditions, i.e.

(i) $u_j^* > 0$, if $g_j(x^*) = 0$.

(ii) $\bigtriangledown g_j(x^*),\ j \in J(x^*, 0)$ are linearly independent.

(iii) $x^T \bigtriangledown^2_{xx} L(x^*, u^*)x > 0$ for all $x \neq 0$ with $\bigtriangledown g_j(x^*)^T x = 0,\ j \in J(x^*, 0)$, where

$$J(x^*, 0) = \{j :\ g_j(x^*) = 0\}. \tag{1.3}$$

This paper is organized as follows. In Section 2 we consider the construction of a hybrid algorithm for solving the large-scale problem (1.1). We discuss the local convergence rate in Section 3. Some numerical results are given in Section 4.

2. Algorithms

In order to solve the equivalent problem (**??**), a class of hybrid methods was proposed in [**?**]. In the hybrid methods, a locally convergent method (method II) is combined with a globally convergent one (method I). In every iteration level, either a complete step of method II or a complete step of method I is carried out. Their combination is based on the following general schema of coupling.

Schema of Coupling:

Step 1: Carry out method I and check test I. If test I is satisfied, then go to Step 2, otherwise go to Step 1.

Step 2: Carry out method II and check test II. If test II is satisfied, then go to Step 2, otherwise go to Step 1.

According to the schema, we choose a truncated Newton-type method as method II and a method of multiplier as method I in the hybrid method for solving large-scale problem with sparse case. In the following the subproblem and its truncated solution are discussed.

2.1 Subproblem and its truncated solution

The Newton-type method generates a sequence of the systems of linear equations, that approximate the local behavior of the problem (**??**) at the current iterate (x, u, t)

$$\begin{pmatrix} -H & A \\ A^T & D \end{pmatrix} \begin{pmatrix} \Delta x \\ -- \\ q_j \end{pmatrix}_{j\in J_k} = \begin{pmatrix} \bigtriangledown_x L(x,u) \\ ---------- \\ -(g_j(x)+t_j) \end{pmatrix}_{j\in J_k}, \tag{2.1}$$

where

$$A = (\bigtriangledown g_j(x))_{j\in J_k}, \quad D = \mathrm{diag}(t_j/u_j)_{j\in J_k},$$

J_k is a index set in J. For Newton-type method, the matrix H in (**??**) is chosen as $\bigtriangledown^2_{xx} L(x,u)$. $m_k = \mathrm{card} J_k$ is the cardinal number of J_k. Thus (**??**) is a $(n+m_k, n+m_k)$-system of linear equations. The detailed description of the subproblem (**??**) refers to [**?**].

Since the advantages of the Newton direction are mainly local, there seems to be no justification for requiring an exact solution to (**??**), when the current iteration point is far away from a local minimizer. Hence, we determine a truncated solution of (**??**) by using an iterative method.

A pair $(\Delta x,\ q_j,\ j \in J_k)$ is called an acceptable truncated solution of (**??**) at (x_k, u_k, t_k), if the following inequality holds:

$$\|\gamma_k\| = \left\| \begin{pmatrix} -H_k & A_k \\ A_k^T & D_k \end{pmatrix} \begin{pmatrix} \Delta x \\ -- \\ q_j \end{pmatrix}_{j\in J_k} - \begin{pmatrix} \bigtriangledown_1 L(x_k,u_k) \\ ---------- \\ -(g_j(x_k)+t_j^{(k)}) \end{pmatrix}_{j\in J_k} \right\| < \eta_k. \tag{2.2}$$

Here $\eta_k > 0,\ \eta_k \to 0,$ if $k \to \infty$ and could insure the local convergence rate of the truncated Newton-type method, which is discussed below.

2.2 Truncated Newton-type method

In order to propose a locally convergent method, the truncating strategy is added to a Newton-type method developed in [**?**]. In addition, we correct $u_j^{(k)}$ if $u_j^{(k)} < \epsilon$ and $t_j^{(k)} < \epsilon'$ (see following remark (i)). In the following we give a truncated Newton-type method.

Algorithm 1

Step 0: Choose $\epsilon,\ \epsilon' > 0,\ x_0 \in R^n$ and $u_0,\ \ t_0 \in R^m$ with $u_j^{(0)} > 0,\ \ t_j^{(0)} > 0,\ j \in J$. Set $k = 0$.

Step 1: If the termination conditions are satisfied, then stop.

Step 2: Solve the subproblem:

2.1) Determine J_k and compute $H_k,\ A_k$ and D_k.

$$\begin{aligned} J_k &= \{j \in J:\ u_j^{(k)} > \epsilon \ \text{ or } \ t_j^{(k)} < \epsilon'\}, \\ H_k &= \nabla^2_{xx} L(x_k, u_k),\ A_k = (\nabla g_j(x_k))_{j\in J_k} \in R^{n\times m_k} \\ D_k &= \mathrm{diag}(t_j^{(k)}/u_j^{(k)})_{j\in J_k} \in R^{m_k\times m_k} \end{aligned}$$

2.2) Determine an acceptable truncated solution Δx_k and q_k, such that

$$\|\gamma_k\| < \min\{1/(k+1)^2,\ 2\Psi(x_k, u_k, t_k)\} \tag{2.3}$$

where

$$\Psi(x_k, u_k, t_k) = \frac{1}{2}(\|P(x_k, u_k, t_k)\|^2 - \sum_{j=1}^{m}(u_j^{(k)} t_j^{(k)})^2 + 4u_k^T t_k), \tag{2.4}$$

$$\gamma_k = \begin{pmatrix} -H_k & A_k \\ A_k^T & D_k \end{pmatrix} \begin{pmatrix} \Delta x \\ -- \\ q_j \end{pmatrix}_{j\in J_k} - \begin{pmatrix} \nabla_x L(x_k, u_k) \\ ---------- \\ -(g_j(x_k) + t_j^{(k)}) \end{pmatrix}_{j\in J_k}.$$

Set

$$q_j^{(k)} = \nabla g_j(x_k)^T \Delta x_k + g_j(x_k) - t_j^{(k)},\ j \in J/J_k\ . \tag{2.5}$$

Step 3: Compute x_{k+1}, u_{k+1} and t_{k+1}.

$$x_{k+1} = x_k + \Delta x_k$$

$$u_j^{(k+1)} = \begin{cases} u_j^{(k)}(1 + q_j^{(k)}/(2u_j^{(k)}))^2 & j \in J_k \\ u_j^{(k)}(q_j^{(k)}/(2t_j^{(k)}))^2 & j \in J/J_k \end{cases} \tag{2.6}$$

$$t_j^{(k+1)} = \begin{cases} t_j^{(k)}(q_j^{(k)}/(2u_j^{(k)}))^2 & j \in J_k \\ t_j^{(k)}(1 + q_j^{(k)}/(2t_j^{(k)}))^2 & j \in J/J_k \end{cases} \tag{2.7}$$

Step 4: Correct u_{k+1}. If $t_j^{(k+1)} < \epsilon'$ and $u_j^{(k+1)} < \epsilon,\ j \in J$, then set $u_j^{(k+1)} = \epsilon$. $k = k+1$, go to Step 1.

Remarks: (i) Because of the assumption A2, it is impossible that $u_j^{(k)}$ and $t_j^{(k)}$ simultaneously approach to zero. Hence the correction in Step 4 is rational. With the definition of J_k, this ensures that the denominators in (**??**) and (**??**) are at least more than $2\min\{\epsilon, \epsilon'\}$. (ii) The condition (**??**) ensures the local convergence rate, which is further explained in Section 3 (see Lemma 3.1). (iii) We determine the truncated solution $(\Delta x_k,\ q_j^{(k)}, j \in J_k)$ by using the conjugate gradient method (see the following remark (iii) of QPFTH algorithm). Because the method needs to compute r_k in each inner iteration, the test of the condition (**??**) does not increase extra computation costs. (iv) In the hybrid method developed in the following, Algorithm 1 is taken as method II.

2.3 A QP-free, truncated hybrid method for solving large-scale problem

In the hybrid method, Algorithm 1 (method II) is combined with a globally convergent method (method I), in which the multiplier penalty function (see [**?**]) is defined by

$$\Phi(x,u,r) = f(x) - \sum_{j\in J(x)} (u_j g_j(x) - 1/2 r_j g_j^2(x)) - 1/2 \sum_{j\in K(x)} u_j^2/r_j, \quad (2.8)$$

where $J(x) = \{j :\ g_j(x) \le u_j/r_j\},\ K(x) = \{1,\dots,m\}/J(x)$. The following method is called QP-free, truncated hybrid method (QPFTH).

QPFTH Algorithm

Step 0: Choose $c_1, c_2, c_3 > 0, \epsilon, \epsilon' > 0, \rho \in (1/4, 1), x_0 \in R^n, u_0 \in R^m$ and $u_j^{(0)} \ge 0,\ j \in J$. Set $r_0 = \{1,\dots,1\}^T \in R^m . k = 0, l = -1$.

$$M_0 = (\| \bigtriangledown_x \Phi(x_0,u_0,r_0)\|^2 + \|b(x_0,u_0,r_0)\|^2)^{\frac{1}{2}}, \quad (2.9)$$

where

$$b_j(x_0,u_0,r_0) = \min(u_j^{(0)}/r_j^{(0)}, g_j(x_0)), j \in J. \quad (2.10)$$

$\tilde{x}_0 = x_0$. Choose or determine a feasible point $x^+ \in R^n$.

Step 1: Carry out multiplier method and check test I.

1.1) Minimize $\Phi(x,u_k,r_k)$ by starting with $\tilde{x}_k$: use a unconstrained minimization method for minimizing $\Phi(x,u_k,r_k)$ such that the function $\Phi(x,u_k,r_k)$ is reduced at each inner iteration. If an inner iteration point $\tilde{x}$ satisfies

$$\| \bigtriangledown_x \Phi(\tilde{x},u_k,r_k)\| \le c_2\eta_k \quad (2.11)$$

$$\Phi(\tilde{x},u_k,r_k) \le f(x^+) + \tfrac{1}{4}(|f(x^+)| + 0.1), \quad (2.12)$$

where $\eta_k = \min(1/(k+1)^2, \rho\| \bigtriangledown_x \Phi(\tilde{x},u_k,r_k)\|, \rho\|b(\tilde{x},u_k,r_k)\|)$, then go to 1.2).

1.2) Compute

$$\bar{M} = \bar{M}(\tilde{x},u_k,r_k) = (\| \bigtriangledown_x \Phi(\tilde{x},u_k,r_k)\|^2 + \|b(\tilde{x},u_k,r_k)\|^2)^{\frac{1}{2}}. \quad (2.13)$$

If $\bar{M} > \rho M_k$, then go to 1.3). Otherwise set $x_{k+1} = \tilde{x},\ r_{k+1} = r_k,\ M_{k+1} = \bar{M}$,

$$u_j^{(k+1)} = u_j^{(k)} - \min(u_j^{(k)}, r_j^{(k+1)} g_j(x_{k+1})),\ j = 1,\dots,m. \quad (2.14)$$

If

$$\min\{g_j(x_{k+1}) :\ j \in J\} \ge -c_1/(k+1)^3 \quad (2.15)$$

is satisfied, then set

$$t_j^{(k+1)} = \begin{cases} g_j(x_{k+1}), & \text{if } g_j(x_{k+1}) > u_j^{(k)}/r_j^{(k+1)} \\ 0, & \text{otherwise} \end{cases} . \quad (2.16)$$

$k = k+1,\ l = k$, go to Step 2.

If (**??**) is not satisfied, then

$$r_j^{(k+1)} = 4r_j^{(k)}, \quad \text{for} \quad j:\ g_j(x) < c_1/(k+1)^3.$$

$k = k+1$, go to 1.1).

1.3) Set $r_j^{(k)} = 10r_j^{(k)}$, for $j:\ |\min(u_j^{(k)}/r_j^{(k)}, g_j(\tilde{x}))| > \frac{1}{4\sqrt{m}}\|b(x_k, u_k, r_k)\|,\ \tilde{x}_k = \tilde{x}$, go to 1.1).

Step 2: Carry out method II (Algorithm 1) and check test II.

2.1) If $t_j^{(k)} < \epsilon'$ and $u_j^{(k)} < \epsilon,\ j \in J$, then set $u_j^{(k)} = \epsilon$. $J_k = \{j \in J:\ u_j^{(k)} > \epsilon \ \text{ or } \ t_j^{(k)} < \epsilon'\}$.

Compute $\Delta x_k \in R^n$ and $q_k \in R^m$ by solving the inequality (**??**) and computing (**??**). Determine $x_{k+1},\ u_{k+1}$ and t_{k+1} according to (**??**) and (**??**) (see Step 2, Step 3 in Algorithm 1). $z_{k+1} = (x_{k+1}, u_{k+1}, t_{k+1})$.

2.2) Test II. If the following conditions are satisfied:

$$\|z_{k+1} - z_k\| \le c_1(1/2)^{k-l+1}, \tag{2.17}$$

$$\max\{|\frac{q_j^{(k)}}{\sqrt{2u_j^{(k)}}}|,\ |1 + \frac{q_j^{(k)}}{2u_j^{(k)}}|\sqrt{2t_j^{(k)}}\} \le c_1(1/2)^{k-l+1},\ j \in J_k, \tag{2.18}$$

$$\max\{|\frac{q_j^{(k)}}{\sqrt{2t_j^{(k)}}}|,\ |1 + \frac{q_j^{(k)}}{2t_j^{(k)}}|\sqrt{2u_j^{(k)}}\} \le c_1(1/2)^{k-l+1},\ j \in J/J_k, \tag{2.19}$$

then

(i) if $k = l$, correct u_k and t_k according to the following rule:

set $u_j^{(k)} = c_3(1/2)^{k+1},\ j \in J$. if $u_j^{(k)} = 0$;
set $t_j^{(k)} = c_3(1/2)^{k+1},\ j \in J$, if $t_j^{(k)} = 0$,

for $j \in J$.

(ii) Set $k = k+1$ and go to 2.1).

Otherwise, go to 2.3).

2.3) Compute $\bar{u}_{k+1}$. Set $\bar{u}_{k+1} = u_{k+1}$. If $g_j(x_{k+1}) > 2u_j^{(k+1)} > 0$, then $\bar{u}_j^{(k+1)} = 0$; if $r_j^{(l)} g_j(x_{k+1}) < -u_j^{(k+1)}$, then $\bar{u}_j^{(k+1)} = \max\{u_j^{(k+1)}, c_3(1/2)^{k+1}\}$. Set

$$\hat{r}_j^{(k+1)} = \begin{cases} -\bar{u}_j^{(k+1)}/g_j(x_{k+1}), & \text{if } r_j^{(l)} g_j(x_{k+1}) < -\bar{u}_j^{(k+1)} \\ \bar{u}_j^{(k+1)}/g_j(x_{k+1}), & \text{if } r_j^{(l)} g_j(x_{k+1}) > \bar{u}_j^{(k+1)} > 0 \\ r_j^{(l)}, & \text{otherwise} \end{cases}, \tag{2.20}$$

$$\hat{u}_j^{(k+1)} = \begin{cases} \bar{u}_j^{(k+1)} + \hat{r}_j^{(k+1)} g_j(x_{k+1}) & \text{if } \hat{r}_j^{(k+1)} g_j(x_{k+1}) \le \bar{u}_j^{(k+1)} \\ 0, & \text{otherwise} \end{cases}. \tag{2.21}$$

$u_{k+1} = \hat{u}_{k+1},\ r_{k+1} = \hat{r}_{k+1},\ k = k+1$, go to Step 1.

Remarks:

(i) In the Step 1.1), many unconstrained minimization technique can be used for minimizing $\Phi(x, u_k, r_k)$. For the large-scale sparse problem, a limited memory BFGS method will be chosen.

(ii) In order to guarantee the numerical stability, we determine the truncated solution $(\Delta x_k,\ q_j^{(k)}, j \in J_k)$ by solving the problem

$$(Q_k^T Q_k + \omega I) \begin{pmatrix} \Delta x \\ -- \\ q_j \end{pmatrix}_{j\in J_k} = Q_k^T \begin{pmatrix} \bigtriangledown_x L(x_k, u_k) \\ ---------- \\ -(g_j(x_k) + t_j^{(k)}) \end{pmatrix}_{j\in J_k}, \qquad (2.22)$$

in stead of the problem

$$Q_k \begin{pmatrix} \Delta x \\ -- \\ q_j \end{pmatrix}_{j\in J_k} = \begin{pmatrix} \bigtriangledown_x L(x_k, u_k) \\ ---------- \\ -(g_j(x_k) + t_j^{(k)}) \end{pmatrix}_{j\in J_k}, \qquad (2.23)$$

where ω is a variable damping factor and

$$Q_k \ = \ \begin{pmatrix} -H_k & A_k \\ A_k^T & D_k \end{pmatrix}.$$

If Q_k is nonsingular and $\omega = 0$, then the problem (**??**) is equivalent to the problem (**??**). If the condition

$$\|Q_k \begin{pmatrix} \Delta x_{k,i} \\ --- \\ q_{j,i} \end{pmatrix}_{j\in J_k} \|^2 > 10^{-10} \| \begin{pmatrix} \Delta x_{k,i} \\ --- \\ q_{j,i} \end{pmatrix}_{j\in J_k} \|^2 \qquad (2.24)$$

is satisfied, then set $\omega_{k,i} = 0$. Otherwise, set $\omega_{k,i_1} = \|Q_k\|_F$, where i_1 is the first index such that (**??**) is not satisfied, and

$$\omega_{k,i} = \begin{cases} 1.5\omega_{k,i-1} & \text{if } \psi_i < \psi_{i-1} \\ 0.5\omega_{k,i-1} & \text{otherwise} \end{cases}$$

for $i > i_1$. Here

$$\psi_i = \|Q_k \begin{pmatrix} \Delta x_{k,i} \\ --- \\ q_{j,i} \end{pmatrix}_{j\in J_k} - \begin{pmatrix} \bigtriangledown_x L(x_k, u_k) \\ ---------- \\ -(g_j(x_k) + t_j^{(k)}) \end{pmatrix}_{j\in J_k} \|,$$

$\Delta x_{k,i},\ q_{j,i}, j \in J_k$ and $\omega_{k,i}$ are the i-th inner iterates in solving the subproblem (**??**) by the conjugate gradient method. In addition, we choose a diagonal matrix $S_k = \text{diag}\{s_{11}^{(k)}, \ldots, s_{\tilde{n},\tilde{n}}^{(k)}\}$ as the scaling matrix of the variables $(\Delta x,\ q_j, j \in J_k)$, where

$$s_{ii}^{(k)} = \begin{cases} \frac{1}{t_i} & \text{if } \ t_i > \kappa \\ 1 & \text{otherwise} \end{cases}$$

for $i = 1, \ldots, \tilde{n},\ \tilde{n} = n + \text{card}J_k,\ t_i$ is the Euclidean norm of the i-th column of M_k, while κ is a small positive constant number. The detailed description of diagonal scaling refers to (Section 2.4.2, [**?**]).

(iii) In Step 2, if method II is carried out again, $u_j^{(l+1)} = 0$ or $t_j^{(l+1)} = 0$ should be avoided, since in this situation we have $u_j^{(k)} = 0$ or $t_j^{(k)} = 0$ for all successive iterations, respectively. Hence, in Step 2.2), we insert some perturbation for $u_j^{(l+1)} = 0$ or $t_j^{(l+1)} = 0$.

(iv) If the starting point x_0 is a feasible point, then set $x^+ = x_0$. Otherwise the feasible point x^+ can be computed by apply a barrier method to the auxiliary R^{n+1} problem

$$\begin{aligned} \min \quad & y \\ \text{subject to} \quad & g_j(x) + y \geq 0,\ j \in J, \end{aligned}$$

where for given $\bar{x} \in R^n$ a start with $\bar{x}$ and $\bar{y} = -2\min\{0, \min\{g_j(x),\ j \in J\}\}$ is possible.

(v) Method I and Method II are used for descreasing M_k and $\Psi(x_k, u_k, t_k)$, respectively. $m_k = 0$ or $\Psi(x_k, u_k, t_k) = 0$ mean that a Kuhn-Tucker pair of (1.1) is obtained. The correction of t_k in Step 1.2 and the correction of $u_k,\ r_k$ in Step 2.3 ensure that M_k and $\Psi(x_k, u_k, t_k)$ are reduced in Steps 1 and 2, respectively. The detailed discussion refers to the proof of global convergence of this algorithm.

In the following section we investigate the local convergence rate.

3. Local Convergence Rate

The convergence rate of QPFTH is determined by that of Algorithm 1. In order to investigate the convergence rate, we consider another problem, which is followed from the problem (**??**). We replace $u,\ t$ in (**??**) by v and y according to the relation

$$u_j = v_j^2/2,\ t_j = y_j^2/2,\ j \in J \tag{3.1}$$

and obtain

$$0 = F(x, v, y) = \left[\begin{array}{c} \bigtriangledown f(x) - 1/2 \sum_{j=1}^m v_j^2 \bigtriangledown g_j(x) \\ - - - - - - - - - - - - - - - - - \\ -g_j(x) + y_j^2/2 \\ - - - - - - - - - - - - - - - - - \\ v_j y_j \end{array} \right]_{j \in J}. \tag{3.2}$$

Because (**??**) is a standard system of nonlinear equations, the convergence analysis developed for the class of problems in the past is directly applied to the system. Hence, first we discuss the problem (**??**), then extended the results to the problem (**??**) and Algorithm 1. For the theoretical analysis, the following algorithm is given for solving the problem (**??**).

Algorithm 1a

Step 0: Choose $\epsilon,\ \epsilon' > 0,\ x_0 \in R^n$ and $v_0,\ y_0 \in R^m$ with $v_j^{(0)} \neq 0,\ y_j^{(0)} \neq 0,\ j \in J$. Set $k = 0$.

Step 1: If the termination conditions are satisfied, then stop.

Step 2: Solve the subproblem.

2.1) Determine J_k and compute $H(x_k, v_k),\ A_k,\ D_k$ and l_k.

$$\begin{aligned} J_k &= \{j \in J : (v_j^{(k)})^2/2 > \epsilon \ \text{ or } \ (y_j^{(k)})^2/2 < \epsilon'\}, \\ H(x_k, v_k) &= \bigtriangledown^2 f(x_k) - 1/2 \sum_{j=1}^{m} v_j^{(k)2} \bigtriangledown^2 g_j(x_k), \qquad (3.3) \\ A_k &= (\bigtriangledown g_j(x_k))_{j \in J_k} \in R^{n \times m_k}, \\ l_k &= \bigtriangledown f(x_k) - 1/2 \sum_{j=1}^{m} v_j^{(k)2} \bigtriangledown g_j(x_k), \qquad (3.4) \\ D_k &= \text{diag}[(y_j^{(k)}/v_j^{(k)})^2]_{j \in J_k}. \end{aligned}$$

2.2) Determine an acceptable truncated solution $\Delta x_k,\ q_k$ by solving the inequality

$$\|\gamma_k\| < \min\{1/(k+1)^2,\ \|F(x_k, v_k, t_k)\|^2\}, \qquad (3.5)$$

where

$$\gamma_k = \begin{pmatrix} -H(x_k, v_k) & A_k \\ A_k^T & D_k \end{pmatrix} \begin{pmatrix} \Delta x \\ - - - \\ q_j \end{pmatrix}_{j \in J_k} - \begin{pmatrix} l_k \\ - - - - - - - - - - - \\ -(g_j(x_k) + (y_j^{(k)})^2/2) \end{pmatrix}_{j \in J_k} \qquad (3.6)$$

Set

$$q_j^{(k)} = \bigtriangledown g_j(x_k)^T \Delta x_k + g_j(x_k) - 1/2(y_j^{(k)})^2, \quad j \in J/J_k.$$

Step 3: Compute $x_{k+1},\ v_{k+1}$ and y_{k+1}.

$$\begin{aligned} x_{k+1} &= x_k + \Delta x_k \\ v_j^{(k+1)} &= \begin{cases} v_j^{(k)}(1 + q_j^{(k)}/(v_j^{(k)})^2) & j \in J_k \\ -v_j^{(k)} q_j^{(k)}/(y_j^{(k)})^2 & j \in J/J_k \end{cases}, \qquad (3.7) \\ y_j^{(k+1)} &= \begin{cases} -y_j^{(k)} q_j^{(k)}/(v_j^{(k)})^2 & j \in J_k \\ y_j^{(k)}(1 + q_j^{(k)}/(y_j^{(k)})^2)) & j \in J/J_k \end{cases}. \qquad (3.8) \end{aligned}$$

Step 4: Correct v_{k+1}. If $(y_j^{(k+1)})^2 < 2\epsilon'$ and $(v_j^{(k+1)})^2 < 2\epsilon,\ j \in J$, then set $v_j^{(k+1)} = \text{sign}(v_j^{(k+1)})\sqrt{2\epsilon}$, $k = k + 1$, go to Step 1.

Remark: Algorithm 1a can be as a perturbed version of Newton's method for solving $F(x, v, y) = 0$:

$$[F'(w_k) - K(w_k)](w_{k+1} - w_k) = -F(w_k) \tag{3.9}$$

with $w_k = (x_k, v_k, y_k)$ and the $(n + 2m, n + 2m)$ perturbation matrix

$$K(w_k) = \begin{pmatrix} 0 & -\tilde{A}(x_k)\mathrm{diag}(\tilde{v}_i^{(k)}) & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{3.10}$$

where $\tilde{A}(x_k) = (\bigtriangledown g_j(x_k))_{j\in J}$ and

$$\tilde{v}_j^{(k)} = \begin{cases} 0 & \text{for } \ j \in J_k \\ v_j^{(k)} & \text{for } \ j \in J/J_k \end{cases} .$$

The relationship between Algorithm 1 and Algorithm 1a is described in the following lemma and the detailed description refers to [**?**].

Lemma 3.1. *Let the sequence* $\{(x_k, u_k, t_k)\}$ *generated by Algorithm* 1 *and the sequence* $\{(x_k, v_k, y_k)\}$ *generated by Algorithm* 1*a. If it holds*

$$u_j^{(0)} = (v_j^{(0)})^2/2, \ t_j^{(0)} = (y_j^{(0)})^2/2, \ j \in J \tag{3.11}$$

then

$$u_j^{(k)} = (v_j^{(k)})^2/2, \ t_j^{(k)} = (y_j^{(k)})^2/2, \ j \in J, \ k = 0, 1, 2, \ldots \tag{3.12}$$

Proof: Assume that (**??**) holds for a nonnegative integer k, we prove that it is true for $k+1$. It is clear that $J_k, \ A_k, \ D_k$ generated by both Algorithms 1 and 1a are same. H_k and $H(x_k, v_k)$ are also same. From (**??**), (**??**) and (**??**), it follows

$$\|F(x_k, v_k, y_k)\|^2 = 2\Psi(x_k, u_k, t_k).$$

Hence, $\Delta x_k, \ q_k$ generated in Step 2.2 are same. From Step 3 of both algorithms, we have, for $j \in J_k$,

$$\begin{aligned} u_j^{(k+1)} &= u_j^{(k)}(1 + q_j^{(k)}/(2u_j^{(k)}))^2 \\ &= 1/2(v_j^{(k)})^2(1 + q_j^{(k)}/(v_j^{(k)})^2)^2 \\ &= 1/2(v_j^{(k)} + q_j^{(k)}/v_j^{(k)})^2 = (v_j^{(k+1)})^2/2. \end{aligned}$$

For $j \in J/J_k$,

$$\begin{aligned} u_j^{(k+1)} &= u_j^{(k)}(q_j^{(k)}/(2t_j^{(k)}))^2 \\ &= 1/2(v_j^{(k)}q_j^{(k)}/(y_j^{(k)})^2)^2 = (v_j^{(k+1)})^2/2. \end{aligned}$$

According to the same computation, we obtain

$$t_j^{(k+1)} = (y_j^{(k+1)})^2/2 \quad \text{for } j \in J.$$

Hence, (**??**) holds for $k+1$. This lemma is proved.

If the coefficient matrix in (**??**) is nonsingular, then an acceptable truncated solution is certainly obtained.

Lemma 3.2. *If*

$$\begin{pmatrix} -H(x_k, v_k) & A_k \\ A_k^T & D_k \end{pmatrix}$$

is nonsingular, then either $(x_k,\ (v_j^{(k)})^2, j \in J)$ *is a Kuhn-Tucker pair of* (1.1) *or an acceptable truncated solution is determined. Moreover, this conclusion is suitable to Algorithm* 1*, i.e. if*

$$\begin{pmatrix} -H_k & A_k \\ A_k^T & D_k \end{pmatrix}$$

in (**??**) *is nonsingular, then either* (x_k, u_k) *is a Kuhn-Tucker pair of* (1.1) *or an acceptable truncated solution in Algorithm 1 is determined.*

Proof: According to the assumption in this lemma, there exists a unique exact solution $\Delta x_{k,*}$ and $q_{j,*}^{(k)},\ j \in J_k$ such that

$$\begin{pmatrix} -H(x_k, v_k) & A_k \\ A_k^T & D_k \end{pmatrix} \begin{pmatrix} \Delta x_{k,*} \\ --- \\ q_{j,*}^{(k)} \end{pmatrix}_{j\in J_k} = \begin{pmatrix} l_k \\ ---------- \\ -(g_j(x_k) + (y_j^{(k)})^2/2) \end{pmatrix}_{j\in J_k}, \tag{3.13}$$

where $H(x_k, v_k)$ and l_k are same as those in (**??**) and (**??**). We consider two cases.

(1) $F(x_k, v_k, y_k) = 0$. Hence, $(x_k, 1/2(v_j^{(k)})^2)_{j\in J}$ is a Kuhn-Tucker pair of (1.1). This means that Algorithm 1a terminates at (x_k, v_k, y_k).

(2) $F(x_k, v_k, y_k) \neq 0$. Then, in finite number of iteration, we can obtain an acceptable truncated solution $(\Delta x_k,\ q_j^{(k)}, j \in J_k)$ such that

$$\|\gamma_k\| \leq \min\{1/(k+1)^2, \|F(x_k, v_k, y_k)\|^2\}, \tag{3.14}$$

where

$$\gamma_k = \begin{pmatrix} -H(x_k, v_k) & A_k \\ A_k^T & D_k \end{pmatrix} \begin{pmatrix} \Delta x_k \\ --- \\ q_j^{(k)} \end{pmatrix}_{j\in J_k} - \begin{pmatrix} l_k \\ ---------- \\ -(g_j(x_k) + (y_j^{(k)})^2/2) \end{pmatrix}_{j\in J_k}.$$

From Lemma 3.1, we obtain that this conclusion in (1) and (2) is also suitable to Algorithm 1.

The following lemma denotes the connection of the convergence properties between two sequences, $\{(x_k, u_k, t_k)\}$ and $\{(x_k, v_k, y_k)\}$.

Lemma 3.3. *Let $\{w_k\} = \{(x_k, v_k, y_k)\}$ and $\{z_k\} = \{(x_k, u_k, t_k)\}$ be sequences satisfing (??) for all k. Let $w^* = (x^*, v^*, y^*)$ and $z^* = (x^*, u^*, t^*)$. w^* and z^* satisfy the relation*

$$u_j^* = (v_j^*)^2/2,\ t_j^* = (y_j^*)^2/2,\ j \in J \tag{3.15}$$

Then it holds:

(a) There are numbers $\delta > 0,\ c > 0$ such that for $w_k \in B(w^*, \delta)$

$$\|w_k - w^*\|^2 \le \|z_k - z^*\|(\delta + 2\sqrt{2m}), \tag{3.16}$$

$$\|z_k - z^*\| \le c\|w_k - w^*\|. \tag{3.17}$$

(b) $\lim_{k\to\infty} w_k = w^*$ implies $\lim_{k\to\infty} z_k = z^*$.

(c) If w_k is Q-quadratically convergent to w^* then $\{z_k\}$ converges two-step Q-quadratically and R-quadratically.

The proof of this lemma refers to [?], while the definitions of Q-quadratic and R-quadratic refer to (Chapter 8, [?]). In order to discuss the convergence of Algorithm 1a, we require following two lemmas.

Lemma 3.4. *Under the condition of assumption A2, the Jacobian of F*

$$F'(x, v, y) = \begin{pmatrix} H(x, v), & -\tilde{A}(x)\,diag(v_j), & 0 \\ -\tilde{A}(x)^T & 0 & diag(y_j) \\ 0 & diag(y_j) & diag(v_j) \end{pmatrix} \tag{3.18}$$

is nonsingular at the point (x^*, v^*, y^*), where $\tilde{A}(x) = (\bigtriangledown g_j(x))_{j\in J}$ and $H(x, v)$ is the same as in (??). Moreover,

$$M^* = \begin{pmatrix} -H(x^*, v^*) & A^* \\ (A^*)^T & 0 \end{pmatrix}$$

is also nonsingular, where

$$A^* = (\bigtriangledown g_j(x^*))_{j\in J(x^*,0)},$$

and $J(x^*, 0)$ refers to (??).

Proof: The first conclusion in this lemma refers to [?]. Let $m^* = \mathrm{card}J(x^*, 0),\ x_1 \in R^n,\ x_2 \in R^{m^*}$. It will be shown that

$$M^* \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 0 \ \text{ implies } \ \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 0.$$

From $M^* \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 0$, we obtain

$$\begin{cases} (A^*)^T x_1 = 0 \\ -H(x^*, v^*)x_1 + A^* x_2 = 0 \end{cases}.$$

The condition (iii) in A2 shows that $x_1 = 0$, while condition (ii) in A2 insures that $x_2 = 0$. Hence M^* is nonsingular.

Lemma 3.5. *Let (x^*, u^*) be a Kuhn-Tucker pair of* (1.1) *and let $z^* = (x^*, u^*, t^*)$ and $w^* = (x^*, v^*, y^*)$ be related solutions of (??) and (??) respectively, where the relation between z^* and w^* satisfies (??). Let the assumption* A2 *be satisfied and assume that the constants ϵ and ϵ' are chosen such that*

$$0 < \quad \epsilon \quad < 1/8 \min\{(v_j^*)^2 : \ j \in J(x^*, 0)\} = 1/4 \min\{u_j^* : \ j \in J(x^*, 0)\} \tag{3.19}$$

$$0 < \quad \epsilon' \quad < 1/8 \min\{(y_j^*)^2 : \ j \in J/J(x^*, 0)\} = 1/4 \min\{t_j^* : \ j \in J/J(x^*, 0)\} \tag{3.20}$$

Then there exists a number $0 < \rho \leq \min\{\sqrt{2\epsilon}, \sqrt{2\epsilon'}\}$ such that

$$J_k = J(x^*, 0) \quad \textit{for all } w_k \in B(w^*, \rho).$$

Although J_k is a little different from I_k in [?], the proof of this lemma is similar to that in [?]. Now we prove the perturbed property of Algorithm 1a.

Theorem 3.6. *Let the system (??) at w_k be solved, w_{k+1} be generated by Algorithm 1a and $\Delta x_k, q_k$ be its acceptable truncated solution. Define $\theta_k \in R^{n+2m}$ by*

$$[F'(w_k) - K(w_k)](w_{k+1} - w_k) = -F(w_k) - \theta_k. \tag{3.21}$$

Then θ_k satisfies

$$\begin{aligned} \theta_j^{(k)} &= \gamma_j^{(k)} \quad j = 1, \ldots, n, \\ \theta_{n+j}^{(k)} &= \begin{cases} \gamma_{n+j}^{(k)} & j \in J_k \\ 0 & j \in J/J_k \quad \textit{or} \quad j > m \end{cases}, \end{aligned}$$

i.e. $\|\theta_k\| = \|\gamma_k\|$, where γ_k is defined in Step 2.2 *of Algorithm* 1a. *$F(w_k), K(w_k)$ and $F'(w_k)$ refer to (??), (??) and (??).*

Proof: From (??) we obtain

$$q_j^{(k)} = v_j^{(k)}(v_j^{(k+1)} - v_j^{(k)}) \ \ j \in J_k.$$

From (??), (??) and (??) it follows

$$\begin{aligned} & (\gamma_j^{(k)})_{j \in \{1, \ldots, n\}} \\ = \ & -H(x_k, v_k)\Delta x_k + A_k (q_j^{(k)})_{j \in J_k} - l_k \\ = \ & -H(x_k, v_k)\Delta x_k + \tilde{A}(x_k)\mathrm{diag}(v_j^{(k)})\Delta v_k - \tilde{A}(x_k)\mathrm{diag}(\tilde{v}_j^{(k)})\Delta v_k - l_k \\ = \ & -[(H(x_k, v_k), -\tilde{A}(x_k)\mathrm{diag}(v_j^{(k)}), 0) - EK(w_k)](\Delta x_k, \Delta v_k, \Delta y_k)^T - l_k \\ = \ & (\theta_j^{(k)})_{j \in \{1, \ldots, n\}}. \end{aligned} \tag{3.22}$$

Here $E \in R^{n\times(n+2m)}$ is the matrix whose i-th row is the i-th row of the identity matrix in $R^{(n+2m)\times(n+2m)}$, $\Delta v_k = v_{k+1} - v_k,\ \Delta y_k = y_{k+1} - y_k$. For $j \in J_k$, it implies from (**??**) and (**??**)

$$\begin{aligned}
\gamma_{n+j}^{(k)} &= [A_k^T \Delta x_k + D_k(q_j^{(k)})_{j\in J_k} - (g_j + 1/2 y_j^{(k)})_{j\in J_k}]_j \\
&= \bigtriangledown g_j(x_k)^T \Delta x_k - y_j^{(k+1)} y_j^{(k)} + g_j(x_k) + 1/2(y_j^{(k)})^2 \\
&= \bigtriangledown g_j(x_k)^T \Delta x_k - y_j^{(k)}(\Delta y_k)_j + g_j(x_k) - 1/2(y_j^{(k)})^2 \\
&= \theta_{n+j}^{(k)}. \qquad (3.23)
\end{aligned}$$

For $j \in J/J_k$, we have from Step 2.2),

$$\begin{aligned}
\theta_{n+j}^{(k)} &= -\bigtriangledown g_j(x_k)^T \Delta x_k + y_j^{(k)} \Delta y_j^{(k)} - g_j(x_k) + 1/2(y_j^{(k)})^2 \\
&= -\bigtriangledown g_j(x_k)^T \Delta x_k - g_j(x_k) + 1/2(y_j^{(k)})^2 + q_j^{(k)} = 0 \qquad (3.24)
\end{aligned}$$

It follows from (**??**) and (**??**)

$$\begin{aligned}
\theta_{n+m+j}^{(k)} &= y_j^{(k)}(\Delta v_k)_j + v_j^{(k)}(\Delta y_k)_j + v_j^{(k)} y_j^{(k)} \\
&= y_j^{(k)} q_j^{(k)}/v_j^{(k)} - v_j^{(k)} y_j^{(k)}(1 + q_j^{(k)}/(v_j^{(k)})^2) + v_j^{(k)} y_j^{(k)} \\
&= 0, \qquad j \in J_k, \qquad (3.25)
\end{aligned}$$

$$\begin{aligned}
\theta_{n+m+j}^{(k)} &= y_j^{(k)}(\Delta v_k)_j + v_j^{(k)}(\Delta y_k)_j + v_j^{(k)} y_j^{(k)} \\
&= -y_j^{(k)} v_j^{(k)}(1 + q_j^{(k)}/(y_j^{(k)})^2 + v_j^{(k)} q_j^{(k)}/y_j^{(k)} + v_j^{(k)} y_j^{(k)} \\
&= 0, \qquad j \in J/J_k. \qquad (3.26)
\end{aligned}$$

(**??**), (**??**), (**??**), (**??**), (**??**) mean that this theorem is proved.

In the following theorem, at first we prove the convergence rate of Algorithm 1a, then apply the convergence results to Algorithm 1.

Theorem 3.7. *Let (x^*, u^*) be a Kuhn-Tucker pair of* (1.1) *and let (x^*, u^*, t^*) be a solution of* (1.2)*. Futhermore, let the assumptions A1 and A2 be satisfied, and assume that ϵ and ϵ' be chosen according to* (**??**) *and* (**??**)*. Then there is a number $\delta^* > 0$ such that for every $z_0 = (x_0, u_0, t_0) \in B(z^*, \delta^*)$ with $u_j^{(0)} > 0,\ t_j^{(0)} > 0$ $(j \in J)$. Algorithm* 1 *terminates after a finite number of iterations with $z_k = z^*$ or $\{z_k\}$ converges two-step Q-quadratically and R-quadratically to z^*.*

Proof: (1) At first we deal with Algorithm 1a. Let $w^* = (x^*, v^*, y^*)$ a solution of (**??**) related z^* via (**??**). Let $w_k \in B(w^*, \rho)$ where $0 < \rho_1 \le \rho$ and ρ is given by Lemma 3.5, such that $F'(w_k)$ and

$$\begin{pmatrix} -H(x_k, v_k) & A_k \\ A_k^T & D_k \end{pmatrix}$$

are nonsingular (see Lemma 3.4). With $\tilde{A}(x)$ as in (**??**), we define

$$\mu = \max\{\|\tilde{A}(x)\| : \quad x \in B(x^*, \rho_1)\}.$$

Since $v_j^* = 0$ for $j \in J/J(x^*, 0)$ and $J_k = J(x^*, 0)$ (see Lemma 3.5), we obtain from (**??**)

$$\begin{aligned} & \|[F'(w_k) - K(w_k)] - F'(w_k)\| = \|K(w_k)\| \le \mu \|\mathrm{diag}(\tilde{v}_j^{(k)})\| \\ \le \quad & \mu \|(\ldots, v_j^{(k)} - v_j^*, \ldots)_{j \in J/J(x^*,0)}^T\| \le \mu \|w_k - w^*\|. \end{aligned} \tag{3.27}$$

Since $\|K(w_k)\| \to 0$, as $w_k \to w^*$, the matrix $F'(w_k) - K(w_k)$ of the system (**??**) can be regarded as a so-called consistent approximation of $F'(w_k)$. Let $\rho_2 > 0$ be sufficiently small such that $\rho_2 \le \rho_1,\ F'(w)$ is nonsingular,

$$\|F(w)\| \le \alpha \|w - w^*\| \tag{3.28}$$

$$\|F'(w) - F'(w^*)\| \le L \|w - w^*\| \tag{3.29}$$

$$\|F(w) - F'(w^*)(w - w^*)\| \le \beta \|w - w^*\|^2 \tag{3.30}$$

for $w \in B(w^*, \rho_2)$ and $\alpha,\ L,\ \beta > 0$. We obtain (**??**) from Lemma 3.4, (**??**) and (**??**) from the assumption A1 and ([**?**], 3.2.12).

From (**??**) and Theorem 3.6, it follows

$$\begin{aligned} (F'(w_k) - K(w_k))(w_{k+1} - w^*) \quad = \quad & -\theta_k + [(F'(w_k) - F'(w^*))(w_k - w^*)] \\ & -[F(w_k) - F'(w^*)(w_k - w^*)] + K(w_k)(w^* - w_k). \end{aligned}$$

With $\|\theta_k\| = \|\gamma_k\| \le \|F(w_k)\|^2$, (**??**), (**??**), (**??**) and (**??**), we obtain

$$\|w_{k+1} - w^*\| \le \eta_1 \|w_k - w^*\|^2.$$

Therefore, with Lemma 3.2, it follows that there is a positive number $\rho_3 \le \rho_2$ such that for every $w_0 \in B(w^*, \rho_1)$ Algorithm 1a generates well-defined iterates $w_k = (x_k, v_k, y_k)$. Moreover either Algorithm 1a terminates after a finite number of iterations with $w_k = w^*$ or $\{w_k\}$ converges to w^* at least with the Q-order 2.

(2) According to the same proof as that of Theorem 1 in [**?**], we obtain that there exists a $\delta^* > 0$ such that for every $z_0 = (x_0, u_0, t_0) \in B(z^*, \delta^*)$ with $u_j^{(0)} > 0,\ t_j^{(0)} > 0,\ j \in J$, a w_0 could be chosen and the following holds: w_0 and z_0 satisfy the relation (**??**), and $\|w_0 - w^*\| \le \min\{\rho_3, \delta\}$, where δ is obtained from Lemma 3.3. Therefore, it follows that Algorithm 1a with the initial choice w_0 possesses the properties stated in (1) of this proof.

(3) From the statement of both Lemmas 3.1 and 3.3, we obtain the asserted properties of Algorithm 1. q.e.d.

We will prove the global convergence of QPFTH algorithm in another paper. Moreover, the global convergence analysis shows that the convergence rate of QPFTH algorithm is the same as Algorithm 1.

TRP	n_v	m_c	n	m	v	L	U
TR1	105	148	30	74	10	0.1	10
TR2	142	210	36	105	10	0.1	10
TR3	172	262	40	131	10	0.001	10
TR4	237	376	48	188	10	0.001	10
TR5	251	400	50	200	10	0.01	10
TR6	350	578	60	289	10	0.001	10

Tabel I. Truss Test Problems

TRP	CPU	IT	IP	IE	EP1	EP2
TR1	50	34	23	15	2.47E-5	5.65E-5
TR2	64	38	25	20	8.34E-6	7.58E-5
TR3	76	40	27	19	2.95E-5	4.38E-5
TR4	98	52	32	24	3.08E-5	6.04E-5
TR5	114	47	31	19	4.66E-6	8.75E-5
TR6	136	61	43	23	1.79E-5	2.56E-5

Tabel II. Numerical Results for Truss Test Problems

4. Numerical Results for Medium-Sized Truss Problems

The implementation and performance of the QPFTH algorithm is discussed in this section. All of the problems have been performed on an SGI INDIGO R4000XS workstation in LESC (Laboratory of Scientific and Engineering Computing, ICMSEC) in Beijing. All calculation within the driving programs, test problems and optimization codes are carried out in double precision.

From searching optimal topology of trusses arise some special sparse large-scale problems. In order to ensure that Assumption 2 in Section 1 is satisfied for the truss problem, we consider some perturbation of the objective function and obtain

$$\begin{aligned} \min_{x\in R^n,\lambda\in R,z\in R^m} \quad & \lambda v - f^T x + \textstyle\sum_{i=1}^m z_i + \quad \alpha(x^T x + \lambda^2 + \sum_{i=1}^m z_i^2) \\ \text{s.t.} \quad & z_i \geq (\tfrac{1}{2}x^T A_i x - \lambda)L_i \\ & z_i \geq (\tfrac{1}{2}x^T A_i x - \lambda)U_i, \quad i = 1,\ldots,m. \end{aligned} \tag{4.1}$$

where $A_i \in R^{n\times n}$, $i = 1,\ldots,m$ are sparse positive semi-definite matrices, $f \in R^n$ and $\alpha \in (0,1)$. The detailed description of the problem refers to [**?**]. The termination conditions are chosen

$$\begin{aligned} \max\nolimits_{1\leq j\leq m} |\min(0, g_j(x))| \leq \epsilon_1, \\ \| \bigtriangledown_1 L(x,u)\|_\infty \leq \epsilon_2. \end{aligned} \tag{4.2}$$

In our numerical tests, we let

$$\epsilon_1 = 0.0001, \quad \epsilon_2 = 0.0001.$$

Now QPFTH algorithm is used for solving the problem (**??**). We choose the suitable values of the constants $c_1,\ c_2,\ c_3,\ \rho,\ \epsilon,\ \epsilon'$ and κ in QPFTH algorithm (see Step 0 and Remark (iii)) by

$$c_1 = 0.1,\ c_2 = 1,\ c_3 = 10^{-3},\ \rho = 0.9,\ \epsilon = \epsilon' = 10^{-7},\ \kappa = 0.5.$$

Some medium-sized truss test problems are given in Tabel I. In this table, n_v denotes the number of the variables, and m_c the number of the constraints. $n,\ m,\ v$ are the same as in (**??**). L_i and U_i in (**??**), $i = 1, \ldots, m$, are chosen as the same value L and U, respectively.

Numerical results are given in Tabel II, where

TRP: Truss test problem,

CPU: Execution time in seconds,

IT: Number of the outer iterations,

IP: Number of minimizing the multiplier penalty function (**??**),

IE: Number of solving the system of linear equations (**??**).

EP1 and EP2 refer to (**??**).

According to numerical experience, l in the limited memory BFGS inverse l-update is chosen as 4.

All these test problems are successfully terminated. Although these test problems are medium-sized, they represent some characteristic of a certain kind of large sparse problems. Numerical results showed that QPFTH algorithm can handle large sparse problems.

The detailed implementation and global convergence of the QPFTH algorithm will be discussed in another paper.

Acknowledgment: I would like to thank referee for his helpful comments and suggestions on a previous version of this paper.

References

[1] R.M. Chamberlain, C. Lemarechal, H.C. Pedersen and M.J.D. Powell, The watchdog technique for forcing convergence in algorithms for constrained optimization, *Math. Prog. study*, 16(1982), 1-17.

[2] T.F. Coleman, Large-scale numerical optimization: introduction and overview, CTC91TR85, Cornell Theory Center, Cornell University, Ithaca, New York, 1991.

[3] R.S. Dembo, S.C. Eisenstat and T. Steihaug, Inexact Newton Methods, *SIAM J. Numer. Anal.*, 19(1982), 400-408.

[4] R. Fletcher, Pratical Methods of Optimization, Vol.2, Constrained Optimization, John Wiley & Sons, New York, Toronto, 1981.

[5] S.P. Han, A globally convergent method for nonlinear programming, *JOTA*, 22(1977), 297-309.

[6] J. Heinz, and P. Spellucci, A successful implementation of the Pantoja-Mayne SQP method, *Optimization Methods and Software*, 4(1994), 1-28.

[7] H. Kleinmichel, C. Richter, and K. Schönefeld, On a class of hybrid methods for smooth constrained optimization, *JOTA*, 73(1992), 465-500.

[8] H. Kleinmichel, and K. Schönefeld, Newton-type methods for nonlinearly constrained programming problems – algorithm and theory, *Optimization*, 19(1988), 397-412.

[9] H. Kleinmichel, and K. Schönefeld, Superlinearly convergent optimization methods without solving QP, Preprint 07-11-89, Mathematisches Institut, TU Dresdner, 1989.

[10] Q. Ni, General large-scale nonlinear programming using sequential quadratic programming methods, *Bayreuther Mathematischen Schriften*, 45(1993), 133-236.

[11] J.M. Ortega and W.C. Rheinboldt, Iterative Solution of Nonlinear Equations in Several variables, Academic Press, New York, 1970.

[12] M.J.D. Powell and Y. Yuan, A recursive quadratic programming algorithm that use differential exact penalty function, *Mathmatical Programmming*, 35(1986), 265-278.

[13] K. Schönefeld, A superlinearly and globally convergent optimization method based on exact penalties and Newton-type methods, Preprint 07-24-87, 07-25-87, Mathematisches Institut, TU Dresdner, 1987.

[14] K. Schönefeld, A superlinearly and globally convergent optimization method independent of strict complementary slackness, Preprint 07-16-90, Mathematisches Institut, Tu Dresdnen, 1990.

[15] R.B. Wilson, A simplical algorithm for concave programming, Ph.D.Dissertation, Graduate School of Business Admistration, Harward University, Boston, 1963.