An SQP Feasible Descent Algorithm for Nonlinear Inequality Constrained Optimization Without Strict Complementarity

JIN-BAO JIAN AND CHUN-MING TANG
College of Mathematics and Information Science
Guangxi University 530004, Nanning, P.R. China
jianjb@gxu.edu.cn

(Received June 2003; revised and accepted September 2004)

Abstract—In this paper, a kind of nonlinear optimization problems with nonlinear inequality constraints are discussed, and a new SQP feasible descent algorithm for solving the problems is presented. At each iteration of the new algorithm, a convex quadratic program (QP) which always has feasible solution is solved and a master direction is obtained, then, an improved (feasible descent) direction is yielded by updating the master direction with an explicit formula, and in order to avoid the Maratos effect, a height-order correction direction is computed by another explicit formula of the master direction and the improved direction. The new algorithm is proved to be globally convergent and superlinearly convergent under mild conditions without the strict complementarity. Furthermore, the quadratic convergence rate of the algorithm is obtained when the twice derivatives of the objective function and constrained functions are adopted. Finally, some numerical tests are reported. © 2005 Elsevier Ltd. All rights reserved.

Keywords—Nonlinear Inequality, Constrained Optimization, SQP, Feasible Descent Algorithm, Superlinear Convergence.

1. INTRODUCTION

It is well known that the sequential quadratic programming (SQP) method is one of the most effective algorithms for solving nonlinear constrained optimization problems since it possesses fast convergence, therefore, the SQP method is studied widely and many papers are published, see [1–21]. Most of the early SQP algorithms belong to so-called infeasible methods, i.e., some suitable penalty functions are used as merit functions and the iterative points are not feasible, see [1–11]. In order to overcome the shortcoming of the infeasibility of the iterative points, a class of so-called feasible SQP algorithms are presented, i.e., the iterative points are all feasible, see [12–17]. In [18], Jian, Zhang, and Xue presented an SQP type feasible method for solving inequality constrained optimization, in which, since the quadratic program (QP) must not be...
In [19], Jian improved the SQP feasible method in [18], such that the starting iterative point may be arbitrary. More recently, Jian, Zhang, and Lai [20] presented a fast and feasible algorithm of sequential systems of equations. In this algorithm, a system of equations is introduced to replace the previous QPs solved in the SQP type methods, that is, the feasible direction is yielded by solving a system of equations. However, the superlinear convergence properties of these proposed SQP algorithms (such as [12–20]) depend strictly on the strict complementarity, which is rather strong and difficult for testing. Recently, some new SQP algorithms (see [22–26]) have been presented, the most advantage of these algorithms is that the superlinear convergence properties are still ensured under weaker conditions without the strict complementarity, but it is regretful that these new SQP algorithms are infeasible and nonmonotone.

In this paper, we present a new SQP algorithm for solving a class of nonlinear optimization problems with nonlinear inequality constraints. In the process of the iteration of this algorithm, the search direction is generated by solving only one convex QP and two explicit computation formulas, the iterative points are all feasible and the objective function value is monotone decreasing. Under weaker assumptions without the strict complementarity, the algorithm is proved to possess global convergence, strong convergence and superlinear convergence as well as quadratic convergence. In order to test the numerical effect, some practical examples are solved by the proposed algorithm.

2. ALGORITHM AND ITS PROPERTIES

In this paper, we consider the nonlinear inequality constrained optimization problem

\[
\begin{align*}
\min & \quad f_0(x), \\
\text{s.t.} & \quad f_i(x) \leq 0, \quad i \in I = \{1, \ldots, m\}.
\end{align*}
\]

We denote the feasible set \(X\) of problem (2.1) and the index set \(I^0\) by

\[
X = \{x \in \mathbb{R}^n : f_i(x) \leq 0, \forall i \in I\}, \quad I^0 = \{0, 1, \ldots, m\}.
\]

The following basic hypothesis is necessary in this paper.

ASSUMPTION A1. Functions \(f_j (j \in I^0)\) are all continuously differentiable, and gradient vectors \(\{\nabla f_j (x), j \in I(x)\}\) are linearly independent for each feasible point \(x \in X\), where the active set \(I(x)\) is defined by

\[
I(x) = \{i \in I : f_i(x) = 0\}.
\]

For convenience of discussions, for subset \(J \subseteq I\), the following notation is used throughout this paper.

\[
\begin{align*}
f(x) &= (f_1(x), \ldots, f_m(x))^T, \quad f_j(x) = (f_j(x), j \in J), \\
g_i(x) &= \nabla f_1(x), \quad i \in I^0, \quad g_j(x) = \nabla f_j(x), j \in J. \tag{2.2}
\end{align*}
\]

For a given iterative point \(x^k \in X\), we first yield an \(\varepsilon\)-active constraint subset \(I_k \supseteq I(x_k)\), such that the matrix \(g_{I_k}(x^k)\) is full of column rank by the pivoting operation \(\text{POP}\) A given below.

PIVOTING OPERATION (POP) A.

STEP 1. Choose a parameter \(\varepsilon > 0\).

STEP 2. Compute the \(\varepsilon\)-active constraint subset \(I(x_k, \varepsilon)\) by

\[
I(x_k, \varepsilon) = \{i \in I : -\varepsilon \leq f_i(x^k) \leq 0\}. \tag{2.3}
\]

STEP 3. If \(I(x_k, \varepsilon) = \emptyset\) or \(\det((g_{I(x^k, \varepsilon)}(x^k))^T g_{I(x^k, \varepsilon)}(x^k)) \geq \varepsilon\), set \(I_k = I(x^k, \varepsilon)\), and \(\varepsilon_k = \varepsilon\), stop; otherwise set \(\varepsilon := (1/2)\varepsilon\) and repeat Step 2, where the matrix \(g_{I(x^k, \varepsilon)}(x^k)\) defined by (2.2).

The properties of POP A are described as follows and its proof can be seen in [27].
PROPOSITION 2.1. Let \( x^k \in X \) and suppose that Assumption A1 holds, then, one has

(i) The pivoting operation \( A \) can be finished in a finite number of computations.

(ii) If a sequence \( \{x^k\} \) of points has an accumulation point, then, there is an \( \bar{\varepsilon} > 0 \), such that the parameters \( \varepsilon_k \) generated by POP \( A \) satisfy \( \varepsilon_k \geq \bar{\varepsilon} \) for all \( k \).

Let point \( x^k \in X \) and \( I_k \) be the corresponding subset generated by POP \( A \), similar to the SQP type method, we consider a quadratic program (QP) as follows,

\[
\begin{align*}
\min \quad & g_0 (x^k)^T d + \frac{1}{2} d^T H_k d, \\
\text{s.t.} \quad & f_i (x^k) + g_i (x^k)^T d \leq 0, \quad i \in I_k,
\end{align*}
\]  

where \( H_k \in \mathbb{R}^{n \times n} \) is a positive definite matrix.

Obviously, QP (2.4) always has a feasible solution \( d = 0 \), so the strict convex program (2.4) always has a (unique) solution, and \( d_0 (x^k) \) is a solution of (2.4) if and only if it is a KKT point of (2.4), i.e., there exists a corresponding KKT multiplier \( \lambda^k_{i_k} = (\lambda^k_i, \ i \in I_k) \), such that

\[
\begin{align*}
g_0 (x^k) + H_k d_0 (x^k) + \sum_{i \in I_k} \lambda^k_i g_i (x^k) = 0, \\
f_i (x^k) + g_i (x^k)^T d_0 (x^k) \leq 0, \quad \lambda^k_i \geq 0, \\
\lambda^k_i \left( f_i (x^k) + g_i (x^k)^T d_0 (x^k) \right) = 0, \quad \forall i \in I_k.
\end{align*}
\]  

It is known that the solution \( d_0 (x^k) \) of (2.4) may not be a feasible direction of the feasible set \( X \) at point \( x^k \), so, in order to generate a feasible direction, it must be updated by some suitable technique, for example, solving another QP [12] or a system of linear equations (see [28–30]). In this work, we use an explicit formula to update \( d_0 (x^k) \) as follows.

\[
d (x^k) = d_0 (x^k) - \delta (x^k) N_k (N_k^T N_k)^{-1} e_k,
\]

with \( e_k = (1, \ldots, 1)^T \in \mathbb{R}^{|I_k|} \) and

\[
\begin{align*}
N_k &= g_{I_k} (x^k), \\
\delta (x^k) &= \frac{\|d_0 (x^k)\| \|d_0 (x^k)^T H_k d_0 (x^k)\|}{2 |e_k^T \pi (x^k)| \|d_0 (x^k)\| + 1}, \\
\pi (x^k) &= - (N_k^T N_k)^{-1} N_k^T g_0 (x^k).
\end{align*}
\]

From the KKT conditions (2.5) and formula (2.6), we have the following relations,

\[
\begin{align*}
g_{I_k} (x^k)^T d (x^k) &= N_k^T d (x^k) = N_k^T d_0 (x^k) - \delta (x^k) e_k \leq -f_{I_k} (x^k) - \delta (x^k) e_k, \\
g_0 (x^k)^T d_0 (x^k) &= -d_0 (x^k)^T H_k d_0 (x^k) - (\lambda^k_{I_k})^T N_k^T d_0 (x^k), \\
&= -d_0 (x^k)^T H_k d_0 (x^k) + (\lambda^k_{I_k})^T f_{I_k} (x^k), \\
g_0 (x^k)^T d (x^k) &= g_0 (x^k)^T d_0 (x^k) - \delta (x^k) g_0 (x^k)^T N_k (N_k^T N_k)^{-1} e_k, \\
&= -d_0 (x^k)^T H_k d_0 (x^k) + (\lambda^k_{I_k})^T f_{I_k} (x^k) + \delta (x^k) \pi (x^k)^T e_k, \\
&\leq -d_0 (x^k)^T H_k d_0 (x^k) + (\lambda^k_{I_k})^T f_{I_k} (x^k) + \delta (x^k) \pi (x^k)^T e_k, \\
&\leq -d_0 (x^k)^T H_k d_0 (x^k) + (\lambda^k_{I_k})^T f_{I_k} (x^k) + \frac{1}{2} d_0 (x^k)^T H_k d_0 (x^k) \\
&= -\frac{1}{2} d_0 (x^k)^T H_k d_0 (x^k) + (\lambda^k_{I_k})^T f_{I_k} (x^k) \leq -\frac{1}{2} d_0 (x^k)^T H_k d_0 (x^k).
\end{align*}
\]
So,
\[ g_0(x^k)^T d(x^k) \leq -\frac{1}{2} d_0(x^k)^T H_k d_0(x^k) < 0. \]  

(2.10)

If the iterative \( x^k \) is not a KKT point of problem (2.1), then, \( d_0(x^k) \neq 0 \), furthermore, from formulas (2.8) and (2.10), one can conclude that \( d(x^k) \) is a feasible descent direction of problem (2.1) at feasible point \( x^k \). On the other hand, to overcome the Maratos effect, a suitable “height-order” auxiliary direction must be adopted. In this paper, the following explicitly auxiliary direction \( d_1(x^k) \) is introduced.

\[ d_1(x^k) = -N_k(N_k^T N_k)^{-1} \left( \|d_0(x^k)\| e_k + \tilde{f}_{I_k}(x^k + d(x^k)) \right), \]

(2.11)

where the constant \( \kappa \in (2, 3) \) and vector
\[ \tilde{f}_{I_k}(x^k + d(x^k)) = f_{I_k}(x^k + d(x^k)) - f_{I_k}(x^k) - g_{I_k}(x^k)^T d(x^k). \]  

(2.12)

REMARK. As we see in the subsequent argument for superlinear convergence, the construction of formulas (2.11), (2.12), especially formula (2.12), is a new technique for computing the height-order correction direction \( d_1(x^k) \), and it plays a very important role in avoiding the strict complementarity.

Now, we describe the steps of our algorithm as follows.

ALGORITHM A.

STEP 0. INITIALIZATION. Let parameters \( \varepsilon_{-1} > 0, \beta \in (0, 1), \alpha \in (0, 0.5), \) and choose a starting feasible point \( x^0 \in X \) and a symmetric positive definite matrix \( H_0 \in \mathbb{R}^{n \times n} \); set \( k := 0 \).

STEP 1. Set the starting parameter \( \varepsilon = \varepsilon_{k-1} \), generate an approximately active constraint set \( I_k \) by POP A and let \( e_k \) be the corresponding termination parameter.

STEP 2. SOLVE QP. Solve QP (2.4) to get a (unique) solution \( d_0 = d_0(x^k) \) and the corresponding KKT multiplier vector \( \lambda^k = (\lambda^j_I, j \in I_k) \). If \( d_0 = 0 \), then, \( x^k \) is a KKT point of (2.1) and stop; otherwise, enter Step 3.

STEP 3. GENERATE SEARCH DIRECTIONS. Compute the improved direction \( d^k = d(x^k) \) by formula (2.6) and the height-order auxiliary direction \( d_1^k = d_1(x^k) \) by (2.11).

STEP 4. DO CURVE SEARCH. Compute the step size \( \tau_k \), the first number \( \tau \) of the sequence \( \{1, \beta, \beta^2, \ldots \} \) satisfying

\[ f_0(x^k + \tau d^k + \tau^2 d_1^k) \leq f_0(x^k) + \alpha \tau g_0(x^k)^T d^k, \]

\[ f_j(x^k + \tau d^k + \tau^2 d_1^k) \leq 0, \quad \forall j I. \]

(2.13)

STEP 5. Compute a new symmetric positive definite matrix \( H_{k+1} \), set \( x^{k+1} = x^k + \tau_k d^k + \tau_k^2 d_1^k \) and \( k := k + 1 \), go back to Step 1.

If the solution \( d_0 \) generated at Step 2 equals zero, one knows from the KKT conditions (2.5), that \( x^k \) is a KKT point of problem (2.1); if \( d_0 \neq 0 \), one can conclude, from (2.8) and (2.10), that \( d^k = d(x^k) \) is a feasible descent direction of (2.1) at point \( x^k \), therefore, the curve search (2.13) can stop in a finite number of computations; moreover, the proposed Algorithm A is well defined from (2.8) and (2.10).

3. GLOBAL CONVERGENCE

If the proposed Algorithm A stops at \( x^k \), we know that the iterative point \( x^k \) is a KKT point of problem (2.1), from formula (2.5). In this section, we assume that an infinite sequence \( \{x^k\} \) of points is yielded by Algorithm A, and we will show that every accumulation point \( x^* \) of \( \{x^k\} \) is a KKT point of (2.1). For this purpose, we further assume that the following condition holds.
ASSUMPTION A2. The sequence \( \{H_k\} \) of matrices is uniformly positive definite, i.e., there exist two positive constants \( a \) and \( b \), such that
\[
a \|d\|^2 \leq d^THkd \leq b \|d\|^2, \quad \forall \ d \in \mathbb{R}^n, \ \forall \ k.
\]

In this section, we suppose that \( x^* \) is a given accumulation point of \( \{x^k\} \), therefore, in view of \( I_k \) being a subset of the finite set \( I = \{1, \ldots, m\} \) and taking into account Proposition 2.1, we can assume without loss of generality that there exists an infinite index set \( K \), such that
\[
x^k \to x^*, \quad I_k \equiv I', \quad \forall \ k \in K; \quad \varepsilon_k \geq \varepsilon, \quad \forall \ k.
\]

PROPOSITION 3.1. Suppose that Assumptions A1 and A2 hold. Then,

(i) Matrix \( N_k^T N_* \) is nonsingular with \( N_* = g_\tau(x^*) \), and there exists a constant \( \xi > 0 \), such that
\[
\left\| (N_k^T N_k)^{-1} \right\| \leq \xi, \quad \forall \ k \in K.
\]

(ii) The sequences \( \{d_0^k: k \in K\} \), \( \{d^k: k \in K\} \) and \( \{d_1^k: k \in K\} \) are all bounded.

(iii) \( \lim_{k \in K} d_0^k = \lim_{k \in K} d^k = \lim_{k \in K} d_1^k = 0 \).

PROOF.

(i) From (3.2) and Proposition A, we have
\[
det (N_k^T N_*) = \lim_{k \in K} det (N_k^T N_k) \geq \lim_{k \in K} \varepsilon_k \geq \varepsilon > 0.
\]

So, the first Conclusion (i) follows.

(ii) In view of the fact that \( d^k = -N_k (N_k^T N_k)^{-1} f_{k+1}(x^k) \) is a feasible solution of QP (2.4) and \( d_0^k \) is an optimal solution, we get
\[
go (x^k)^T d_0^k + \frac{1}{2} (d_0^k)^T H_k d_0^k \leq go (x^k)^T d^k + \frac{1}{2} (d^k)^T H_k d^k.
\]

Again, from Part (i) and \( \lim_{k \in K} x^k = x^* \), we know that \( \{g_0(x^k): k \in K\} \) and \( \{d^k: k \in K\} \) are all bounded, i.e., there exists a constant \( \bar{c} > 0 \), such that \( \|g_0(x^k)\| \leq \bar{c} \) and \( \|d^k\| \leq \bar{c} \) for all \( k \in K \), therefore, one has from the inequality above and (3.1),
\[
-\bar{c} \|d_0^k\| + \frac{1}{2} \|d_0^k\|^2 \leq \bar{c}^2 \left( 1 + \frac{1}{2} b \right),
\]

this inequality shows that \( \{d_0^k: k \in K\} \) is bounded. Furthermore, the boundedness of \( \{d^k: k \in K\} \) and \( \{d_1^k: k \in K\} \) is at hand from (2.6), (2.11), and Part (i).

(iii) To prove Part (iii), in view of Result (i) and formulas (2.6) and (2.11), it is sufficient to show \( \lim_{k \in K} d_0^k = 0 \). For this purpose, we suppose by contradiction that \( \lim_{k \in K} d_0^k \neq 0 \), then, there exists an infinite index set \( K' \subseteq K \) and a constant \( \sigma > 0 \), such that \( \|d_0^k\| \geq \sigma \) holds, for \( k \in K' \subseteq K \) large enough. The proof is divided into two steps as follows, and we assume that \( k \in K' \) is sufficiently large and \( \tau > 0 \) is sufficiently small.

(A) Suppose that there exists a constant \( \bar{\tau} > 0 \), such that the step size \( \tau_k \geq \bar{\tau}, \) for \( k \in K' \) large enough.

Analyze the first search inequality of (2.13): using Taylor expansion, combining (2.10) and (3.1), one has
\[
f_0 (x^k + \tau d^k + \tau^2 d_1^k) = f_0 (x^k) + \alpha \tau g_0 (x^k)^T d^k + (1 - \alpha) \tau g_0 (x^k)^T d_0^k + o(\tau)
\]
\[
\leq f_0 (x^k) + \alpha \tau g_0 (x^k)^T d^k - \frac{1}{2} (1 - \alpha) \tau (d_0^k)^T H_k d_0^k + o(\tau)
\]
\[
\leq f_0 (x^k) + \alpha \tau g_0 (x^k)^T d^k - \frac{1}{2} (1 - \alpha) \alpha \tau \|d_0^k\|^2 + o(\tau)
\]
\[
\leq f_0 (x^k) + \alpha \tau g_0 (x^k)^T d^k - \frac{1}{2} (1 - \alpha) \alpha \sigma^2 + o(\tau).
\]
The last inequality shows that the first inequality of (2.13) holds, for \( k \in K' \) large enough and \( \tau > 0 \) small enough.

Analyze the second inequality of (2.13). If \( j \notin I(x^*) \), i.e., \( f_j(x^*) < 0 \), from the continuity of function \( f_j(x) \) and the boundedness of \( \{d^k, d_t^k : k \in K'\} \), we know \( f_j(x^k + \tau d^k + \tau^2 d_t^k) \leq 0 \) holds, for \( k \in K' \) large enough and \( \tau > 0 \) small enough.

Let \( j \in I(x^*) \), i.e., \( f_j(x^*) = 0 \), then, \( j \in I_k \) by Proposition 2.1(ii), similarly, using Taylor expansion and (2.8), we have

\[
 f_j (x^k + \tau d^k + \tau^2 d_t^k) = f_j (x^k) + \tau g_j (x^k)^T d^k + o(\tau) \\
 \leq f_j (x^k) - \tau f_j (x^k) - \tau \delta (x^k) + o(\tau).
\]

On the other hand, formula (2.7) gives

\[
 \delta (x^k) = \frac{(d_t^k)^T H_k d_t^k}{2 \|e_1^k \pi (x^k)\| + 1/\|d_t^0\|} \geq \frac{a \|d_t^k\|^2}{2 \|e_1^k \pi (x^k)\| + 1/\sigma} \geq a \|d_t^k\|^2 \geq a_0^2.
\]

Thus,

\[
 f_j (x^k + \tau d^k + \tau^2 d_t^k) \leq (1 - \tau) f_j (x^k) - \tau \bar{a} \sigma^2 + o(\tau) \leq 0
\]

holds, for \( k \in K' \) large enough and \( \tau > 0 \) small enough.

Summarizing the analysis above, we conclude that there exists a \( \bar{\tau} > 0 \), such that \( \tau_k \geq \bar{\tau} \), for all \( k \in K' \).

\begin{itemize}
  \item \textbf{(B)} Use \( \tau_k \geq \bar{\tau} > 0 \) to bring a contraction. From the first inequality of (2.13), (2.10), and (3.1), we have
    \[
    f_0 (x^{k+1}) \leq f_0 (x^k) + \alpha \tau_k g_0 (x^k)^T d^k \\
    \leq f_0 (x^k) - \frac{1}{2} \alpha \tau_k (d_t^k)^T H_k d_t^k \\
    \leq f_0 (x^k) - \frac{1}{2} \alpha \tau_k \|d_t^0\|^2, \quad \forall \ k.
    \]

    This shows that \( \{f_0(x^k)\} \) is decreasing, combining \( \lim_{k \in K} f_0 (x^k) = f_0 (x^*) \), one knows \( \lim_{k \to \infty} f_0 (x^k) = f_0 (x^*) \). On the other hand, one also has
    \[
    f_0 (x^{k+1}) \leq f_0 (x^k) - \frac{1}{2} \alpha \bar{\tau} \sigma^2, \quad \forall \ k \in K'.
    \]

    Passing to the limit \( k \in K' \) and \( k \to \infty \) in this inequality, we have \((-1/2)\alpha \bar{\tau} \sigma^2 \geq 0\), which is a contradiction, and the whole proof is completed.
\end{itemize}

**THEOREM 3.1.** Suppose that Assumptions A1 and A2 hold, then, Algorithm A either stops at a KKT point \( x^k \) of problem (2.1) in a finite number of steps or generates an infinite sequence \( \{x^k\} \) of points, such that each accumulation point \( x^* \) is a KKT point of problem (2.1). Furthermore, there exists an index set \( K \), such that \( \{(x^k, \lambda^k) : k \in K\} \) converges to the KKT pair \( (x^*, \lambda^*) \), where \( \lambda^k = (\lambda^k_{x^k}, 0_{I \setminus I_k}) \).

**PROOF.** From the KKT condition (2.5), we have

\[
 g_0 (x^k) + H_k d_t^k + N_k \lambda^k_{x^k} = 0.
\]

This together with Proposition 3.1 shows that

\[
 \lambda^k_{x^k} = \lambda^*_x = - (N_k^T N_k)^{-1} N_k^T (g_0 (x^k) + H_k d_t^k) \rightarrow - (N^*_x N_x)^{-1} N^*_x g_0 (x^*) \overset{\text{def}}{=} \lambda^*_x, \quad k \in K.
\]

Therefore, passing to the limit \( k \in K \) and \( k \to \infty \) in (2.5), one has

\[
 g_0 (x^*) + N_x \lambda^*_x = 0, \quad f_{\lambda^*_x} (x^*) \leq 0, \quad \lambda^*_x \geq 0, \quad f_{\lambda^*_x} (x^*)^T \lambda^*_x = 0.
\]

This relationship shows that \( (x^*, \lambda^*) \) with \( \lambda^* = (\lambda^*_x, 0_{I \setminus I'}) \) is a KKT point of problem (2.1), and the proof is completed. \( \blacksquare \)
4. STRONG AND SUPERLINEAR CONVERGENCE

In this section, we will discuss the strong convergence and superlinear property of the proposed algorithm under some mild conditions without the strict complementarity, for this, the following further hypothesis is necessary.

**Assumption A3.**

(i) The functions \( f_j(x) \) \((j \in I^0)\) are all twice continuously differentiable in the feasible set \( X \).

(ii) The sequence \{\( x^k \)\} generated by Algorithm A is bounded, and possess an accumulation point \( x^* \) (so, \( x^* \) is a KKT point from Theorem 3.1), such that the KKT pair \((x^*, \lambda^*)\) satisfies the strong second-order sufficiency conditions, i.e.,

\[
\begin{align*}
d^T \nabla^2 x L(x^*, \lambda^*) d & > 0, \quad \forall \ d \in \Omega \triangleq \{d \in \mathbb{R}^n : \ d \neq 0, \ g_{I^+}(x^*)^T d = 0\},
\end{align*}
\]

where

\[
L(x, \lambda) = f_0(x) + \sum_{j \in I} \lambda_j f_j(x), \quad I^+ = \{j \in I : \lambda_j^* > 0\}.
\]

**Theorem 4.1.**

(i) Suppose that Assumptions A1 and A2 hold, and \( \{x^k\} \) is bounded, then,

\[
\lim_{k \to \infty} d_0^k = \lim_{k \to \infty} d^k = \lim_{k \to \infty} d_1^k = 0 \quad \text{and} \quad \lim_{k \to \infty} \|x^{k+1} - x^k\| = 0.
\]

(ii) If Assumptions A1, A2, and A3 are all satisfied, then, \( \lim_{k \to \infty} x^k = x^* \), and Algorithm A is said to be strongly convergent in this sense.

**Proof.**

(i) Since \( \{x^k\} \) is bounded, from Proposition 3.1(iii), one can conclude that any subsequence \((\{d_0^k, d^k, d_1^k\} : k \in K)\) of \( \{d_0^k, d^k, d_1^k\}_{k=1}^\infty \) must possesses an accumulation point \((0, 0, 0) \in \mathbb{R}^{3n}\), this fact shows that \( \lim_{k \to \infty} (d_0^k, d^k, d_1^k) = (0, 0, 0) \). Moreover, one has

\[
\lim_{k \to \infty} \|x^{k+1} - x^k\| = \lim_{k \to \infty} \|\tau_k d^k + \tau_k^2 d_1^k\| \leq \lim_{k \to \infty} (\|d^k\| + \|d_1^k\|) = 0.
\]

So, Part (i) follows.

(ii) Under the strong second-order sufficiency conditions A3(ii), one can conclude that the given limit point \( x^* \) is an isolated KKT point of (2.1) (see Theorem 1.2.5 in [13] or [31]), therefore \( x^* \) is an isolated accumulation point of \( \{x^k\} \) from Theorem 3.1, and this, together with \( \lim_{k \to \infty} \|x^{k+1} - x^k\| = 0 \), shows that \( \lim_{k \to \infty} x^k = x^* \) (see Theorem 1.1.5 in [13] or [31]).

The proof is finished.

**Lemma 4.1.** Suppose that Assumptions A1, A2, and A3 hold, then,

\[
\|d^k\| \sim \|d_0^k\|, \quad \|d^k - d_0^k\| = O(\|d_0^k\|^3), \quad \|d_1^k\| = O(\|d_0^k\|^2), \quad \|d_1^k\| = O(\|d_0^k\|^2),
\]

\[
I^+ \subseteq J_k \triangleq \{i \in I_k : f_i(x^k) + g_i(x^k)^T d_0^k = 0\} \subseteq I(x^*) \subseteq I_k. \tag{4.1}
\]

**Proof.** The proof of (4.1) is elementary from (2.6), (2.11), and Proposition 3.1(i). To show relationship (4.2), one first gets \( J_k \subseteq I(x^*) \subseteq I_k \) from \( \lim_{k \to \infty} (x^k, d_0^k) = (x^*, 0) \) and Proposition 2.1(ii). Furthermore, one has \( \lim_{k \to \infty} \lambda^*_{I^+} > 0 \) from Theorem 3.1, so \( \lambda^*_{I^+} > 0 \) and \( I^+ \subseteq J_k \) holds, for \( k \) large enough.

To assure the step size \( \tau_k \equiv 1 \), for \( k \) large enough, an additional assumption as follows is necessary.
ASSUMPTION A4. Suppose that $(\nabla^2_{xx} L(x^k, \lambda^k_h) - H_k) d_k = o(\|d_k\|)$, where $L(x, \lambda^k_h) = f_0(x) + \sum_{j \in I_k} \lambda^k_j f_j(x)$.

THEOREM 4.2. Suppose that Assumptions A1–A4 hold. Then, the step size in Algorithm A always equals one, i.e., $\tau_k \equiv 1$, if $k$ is sufficiently large.

PROOF. We know that it is sufficient to verify (2.13) holds, for $\tau = 1$ and $k$ large enough, and the statement “$k$ large enough” will be omitted in the following discussions.

We first prove the second inequalities of (2.13) hold, for $\tau = 1$. For $j \notin I(x^*)$, i.e., $f_j(x^*) < 0$, in view of $(x^k, d^k, d^*_j) \to (x^*, 0, 0) \ (k \to \infty)$, we can conclude $f_j(x^k + d^k + d^*_j) \leq 0$ holds.

For $j \in I(x^*) \subseteq I_k$, one has from (2.8) and (2.11)

$$g_j (x^k)^T d^k = g_j (x^k)^T d^*_0 - \delta (x^k),$$

$$g_j (x^k)^T d^*_j = -\|d^*_j\|^2 - f_j (x^k + d^k) + f_j (x^k) + g_j (x^k)^T d^k, \quad j \in I_k.$$

Hence, we have from Taylor expansion, (4.1) and this relationship,

$$f_j (x^k + d^k + d^*_j) = f_j (x^k + d^k) + g_j (x^k + d^k)^T d^*_j + O \left(\|d^*_j\|^2\right)$$

$$= f_j (x^k + d^k) + g_j (x^k)^T d^*_j + O \left(\|d^*_j\| \cdot \|d^k\|\right) + O \left(\|d^k\|^2\right)$$

$$= -\|d^*_j\| + f_j (x^k) + g_j (x^k)^T d^k + O \left(\|d^*_j\|^2\right)$$

$$= -\|d^*_j\| + f_j (x^k) + g_j (x^k)^T d^*_j - \delta (x^k) + O \left(\|d^*_j\|^2\right)$$

$$\leq -\|d^*_j\|^2 + O \left(\|d^*_j\|^3\right) \leq 0, \quad j \in I(x^*).$$

So, this shows that the second inequalities of (2.13) hold for $\tau = 1$ and $k$ large enough.

The next objective is to show the first inequality of (2.13) holds, for $\tau = 1$ and $k$ large enough. From Taylor expansion and taking into account relationship (4.1), we have

$$w_k \overset{\text{def}}{=} f_0 (x^k + d^k + d^*_j) - f_0 (x^k) - \alpha g_0 (x^k)^T d^k$$

$$= g_0 (x^k)^T (d^k + d^*_j) + \frac{1}{2} (d^k)^T \nabla^2 f_0 (x^k) d^k - \alpha g_0 (x^k)^T d^k + o \left(\|d^k\|^2\right).$$

On the other hand, from the KKT condition of (2.4) and the active set $J_k$ defined by (4.2), one has

$$g_0 (x^k) = -H_k d^*_0 - \sum_{j \in J_k} \lambda^k_j g_j (x^k) = -H_k d^*_0 - \sum_{j \in J_k} \lambda^k_j g_j (x^k) + o \left(\|d^k\|^2\right).$$

So, in view of (4.5) and (4.1), we have

$$g_0 (x^k)^T d^k = - (d^k)^T H_k d^k - \sum_{j \in J_k} \lambda^k_j g_j (x^k)^T d^k + o \left(\|d^k\|^2\right)$$

$$= - (d^k)^T H_k d^k - \sum_{j \in J_k} \lambda^k_j g_j (x^k)^T d^*_j + o \left(\|d^k\|^2\right),$$

$$g_0 (x^k)^T (d^k + d^*_j) = - (d^k)^T H_k d^k - \sum_{j \in J_k} \lambda^k_j g_j (x^k)^T (d^k + d^*_j) + o \left(\|d^k\|^2\right).$$

Again, from (4.3), (4.1), and Taylor expansion, we have

$$o(\|d^k\|^2) = f_j (x^k + d^k + d^*_j) = f_j (x^k) + g_j (x^k)^T (d^k + d^*_j) + \frac{1}{2} (d^k)^T \nabla^2 f_j (x^k) d^k + o(\|d^k\|^2), \quad j \in J_k.$$
Thus, we get

\[- \sum_{j \in J_k} \lambda_j^k g_j(x^k)^T (d^k + d_1^k) = \sum_{j \in J_k} \lambda_j^k f_j(x^k) + \frac{1}{2} (d^k)^T \left( \sum_{j \in J_k} \lambda_j^k \nabla^2 f_j(x^k) \right) d^k + o\left(\|d^k\|^2\right).\]

So, (4.7) and (4.8) give

\[g_0(x^k)^T (d^k + d_1^k) = - (d^k)^T H_k d^k + \sum_{j \in J_k} \lambda_j^k f_j(x^k) + \frac{1}{2} (d^k)^T \left( \sum_{j \in J_k} \lambda_j^k \nabla^2 f_j(x^k) \right) d^k + o\left(\|d^k\|^2\right).\]

Substituting (4.9) and (4.6) into (4.4), we have

\[w_k = \left(\alpha - \frac{1}{2}\right) (d^k)^T H_k d^k + (1 - \alpha) \sum_{j \in J_k} \lambda_j^k f_j(x^k) + \frac{1}{2} (d^k)^T \left( \sum_{j \in J_k} \lambda_j^k \nabla^2 f_j(x^k) - H_k \right) d^k + o\left(\|d^k\|^2\right)\]

\[= \left(\alpha - \frac{1}{2}\right) (d^k)^T H_k d^k + (1 - \alpha) \sum_{j \in J_k} \lambda_j^k f_j(x^k) + \frac{1}{2} (d^k)^T \left( \nabla^2 f_0(x^k, \lambda^k) - H_k \right) d^k + o\left(\|d^k\|^2\right).\]

This, together with (3.1) and Assumption A4 as well as \(\lambda_j^k f_j(x^k) \leq 0\), shows that

\[w_k \leq \left(\alpha - \frac{1}{2}\right) \alpha\|d^k\|^2 + o\left(\|d^k\|^2\right) \leq 0.\]

Hence, the first inequality of (2.13) holds, for \(\tau = 1\) and \(k\) large enough. The whole proof is finished.

At the end of this section, based on Theorem 4.2, we can establish the superlinear convergence of the proposed algorithm as follows.

**Theorem 4.3.** Suppose that Assumptions A1–A4 are all satisfied. Then, the given Algorithm A is superlinearly convergent, i.e., \(\|x^{k+1} - x^*\| = o(\|x^k - x^*\|).\)

**Proof.** From Theorem 4.2 and Lemma 4.1, we know that the sequence \(\{x^k\}\) yielded by Algorithm A has the form of

\[x^{k+1} = x^k + d^k + d_1^k = x^k + d_0^k + (d^k - d_0^k + d_1^k) \text{ def } x^k + d_0^k + \bar{d}^k, \text{ for } k \text{ large enough,}\]

where \(d_0^k\) is a solution of the QP (2.4) and \(\bar{d}^k = O(\|d_0^k\|^2)\). Therefore, the proposed Algorithm A is a special case of the Algorithm Model 1.1 in [21], and the conclusion follows immediately from Theorem 2.3 in [21] which established a general result of convergence rate of Algorithm Model 1.1 in [21].

5. QUADRATIC CONVERGENCE

In Section 4, we have discussed the superlinear convergence of the proposed Algorithm A under mild conditions without the strict complementarity. In this section, we further analyse the quadratic convergence of Algorithm A when the matrix \(H_k\) is yielded by the Hessian matrices \(\nabla^2 f_j(x) (j \in I^0)\).
THEOREM 5.1. Assume that Assumptions A1–A3 hold and $f_j \in C^3$ ($j \in I^0$). If the matrix $H_k$ ($k \geq 0$) in Algorithm A is yielded by

\begin{equation}
H_k = \nabla^2_{xx} L (x^k, \lambda^{k-1}_k) = \nabla^2 f_0 (x^k) + \sum_{j \in I} \lambda^k_j \nabla^2 f_j (x^k), \tag{5.1}
\end{equation}

\begin{equation}
\lambda^k_j = \begin{cases} 
\lambda^k_{j-1}, & j \in I_{k-1}, \\
0, & j \in I \setminus I_{k-1}.
\end{cases}
\end{equation}

Then,

(i) Algorithm A is superlinear convergence.

(ii) The sequence $\{(x_k, \lambda_k^{k-1})\}$ converges quadratically to $(x^*, \lambda^*)$, i.e.,

\begin{equation}
\| (x^{k+1}, \lambda^k) - (x^*, \lambda^*) \| = O \left( \| (x^k, \lambda^{k-1}) - (x^*, \lambda^*) \| ^2 \right).
\end{equation}

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

PROOF. First, we know from Theorems 4.1 and 3.1 that $\lim_{k \to \infty} (x_k, \lambda_k^{k-1}) = (x^*, \lambda^*)$, thus, Assumption A4 holds when the matrix $H_k$ is computed by (5.1), therefore, we conclude part (i) follows from Theorem 4.3. Furthermore, from Theorem 4.2, we know that Algorithm A is a special case of Algorithm Model 1.1 in [21], so one can conclude Parts (ii) and (iii) hold true from Theorems 2.6 and 2.7 in [21].

THEOREM 5.2. Suppose that Assumptions A1–A3 hold and $f_j \in C^3$ ($j \in I^0$). If the matrix $H_k$ ($k \geq 1$) in Algorithm A is generated by

\begin{equation}
H_k = \nabla^2_{xx} L (x^k, \mu(x^k)) = \nabla^2 f_0 (x^k) + \sum_{j \in I} \mu_j (x^k) \nabla^2 f_j (x^k), \tag{5.2}
\end{equation}

\begin{equation}
\mu(x^k) = - \left( g_I (x^k)^\top g_I (x^k) + D (x^k) \right)^{-1} g_I (x^k)^\top g_0 (x^k),
\end{equation}

\begin{equation}
D (x^k) = \text{diag} \left( f_j (x^k)^2, j \in I \right).
\end{equation}

Then, Algorithm A is quadratically superlinear, i.e., $\| x^{k+1} - x^* \| = O(\| x^k - x^* \|^2)$.

PROOF. First, from [10], we know that the function $\mu(x)$ defined above is differentiable in the feasible set $X$, $\mu(x^*) = \lambda^*$, and satisfies $\| \mu(x^k) - \mu(x^*) \| = O(\| x^k - x^* \|)$, so Assumption 2.9 given in [21] holds. Notice that Algorithm A is a special case of the Algorithm Model 1.1 in [21], one can conclude that the conclusion holds true from Theorem 2.10 in [21].

6. NUMERICAL RESULTS

In order to test the computation efficiency of the proposed algorithm, some preliminary numerical tests are reported in this part, and the computing results show that Algorithm A is efficient. We wrote a MATLAB code and utilized the optimization toolbox within MATLAB 6.5 to solve the quadratic program (2.4).

As we know, the computing method of matrix $H_k$ is very important in the SQP method, it determines the superlinearly convergent property of the proposed algorithm. So, the first issue to be addressed is to select an updating procedure for $H_k$, there are at least three formulas we can choose (see [21]). The first two formulas are given by (5.1), (5.2), and another one is given below. It is the so-called BFGS formula (see [32]).

\begin{equation}
H_{k+1} = H_k - \frac{H_k s^k (s^k)^\top H_k}{(s^k)^\top H_k s^k} + \frac{\tilde{y}^k (\tilde{y}^k)^\top}{(s^k)^\top \tilde{y}^k} \quad (k \geq 0), \tag{6.1}
\end{equation}
where
\[ s^k = x^{k+1} - x^k, \quad y^k = y^k + a_k (\gamma_k s^k + A_k A_k^T s^k), \quad \gamma_k = \min\{1, \|s^k\|^2, \xi\}, \quad \xi \in (0, 1), \]
\[ y^k = \nabla_x L(x^{k+1}, \lambda^k) - \nabla_x L(x^k, \lambda^k), \quad A_k = (\nabla f_j(x^k), j \in L_k), \]
\[ \nabla_x L(x, \lambda) = \nabla f_0(x) + \sum_{j \in L_k} \lambda_j \nabla f_j(x), \quad L_k = \{j : f_j(x^k) + g_j(x^k)^T d_0(x^k) = 0, j \in L_k\}, \]
\[ a_k = \begin{cases} 0, & \text{if } (s^k)^T y^k \geq \delta \|s^k\|^2, \quad \delta \in (0, 1); \\ 1, & \text{if } 0 \leq (s^k)^T y^k < \delta \|s^k\|^2; \\ 1 + \frac{\gamma_k \|s^k\|^2 - (s^k)^T y^k}{y^k A_k (A_k)^T s^k}, & \text{otherwise}. \end{cases} \]

We select some test problems from [33-35], which were given below (with starting point \(x^0\)). For all the test problems, we numerically compared our algorithm with our foregoing SQP algorithm [18] proposed by Jian, Zhang and Xue. For the convenience of representation, the two algorithms were abbreviated as Algorithm A and Algorithm JZX respectively, in the rest of this paper.

**PROBLEM 1.** Example hs113 [33],
\[
\begin{align*}
\min & \quad f_0(x) = x_1^2 + x_2^2 + x_1 x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 \\
& \quad + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45, \\
\text{s.t.} & \quad f_1(x) = -105 + 4x_1 + 5x_2 - 3x_7 - 9x_8 \leq 0, \\
& \quad f_2(x) = 10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0, \\
& \quad f_3(x) = -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0, \\
& \quad f_4(x) = 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \leq 0, \\
& \quad f_5(x) = 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leq 0, \\
& \quad f_6(x) = 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_3^2 - x_6 - 30 \leq 0, \\
& \quad f_7(x) = x_1^2 + 2(x_2 - 2)^2 - 2x_1 x_2 + 14x_5 - 6x_6 \leq 0, \\
& \quad f_8(x) = -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \leq 0, \\
& \quad x^0 = (2, 3, 5, 5, 1, 2, 7, 3, 6, 10)^T.
\end{align*}
\]

**PROBLEM 2.** Example s264 [34],
\[
\begin{align*}
\min & \quad f_0(x) = x_1^2 + x_2^2 + x_1 x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 \\
& \quad + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45, \\
\text{s.t.} & \quad f_1(x) = -105 + 4x_1 + 5x_2 - 3x_7 - 9x_8 \leq 0, \\
& \quad f_2(x) = 10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0, \\
& \quad f_3(x) = -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0, \\
& \quad f_4(x) = 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \leq 0, \\
& \quad f_5(x) = 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leq 0, \\
& \quad f_6(x) = 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_3^2 - x_6 - 30 \leq 0, \\
& \quad f_7(x) = x_1^2 + 2(x_2 - 2)^2 - 2x_1 x_2 + 14x_5 - 6x_6 \leq 0, \\
& \quad f_8(x) = -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \leq 0, \\
& \quad x^0 = (2, 3, 5, 5, 1, 2, 7, 3, 6, 10)^T.
\end{align*}
\]

**PROBLEM 3.** (Wolfe [1972]),
\[
\begin{align*}
\min & \quad f_0(x) = \frac{4}{3} (x_1^2 - x_1 x_2 + x_2^2)^{1/3} - x_3, \\
\text{s.t.} & \quad f_1(x) = -x_1 \leq 0, \quad f_2(x) = -x_2 \leq 0, \quad f_3(x) = -x_3 \leq 0, \quad f_4(x) = x_3 - 2 \leq 0, \\
& \quad x^0 = (5, 5, 1)^T.
\end{align*}
\]
PROBLEM 4. Example s225 [34],

\[
\begin{align*}
\min & \quad f_0(x) = x_1^2 + x_2^2, \\
\text{s.t.} & \quad f_1(x) = -x_1 - x_2 + 1 \leq 0, \quad f_2(x) = -x_1^2 - x_2^2 + 1 \leq 0, \\
& \quad f_3(x) = -9x_1^2 - x_2^2 + 9 \leq 0, \quad f_4(x) = -x_1^4 + x_2 \leq 0, \quad f_5(x) = -x_1^2 + x_2 \leq 0, \\
& \quad x^0 = (10, 10)^T.
\end{align*}
\]

PROBLEM 5. Example hs100 [33],

\[
\begin{align*}
\min & \quad f_0(x) = (x_1 - 10)^2 + 5 (x_2 - 12)^2 + x_3^4 + 3 (x_4 - 11)^2 \\
& \quad + 10x_5^2 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7, \\
\text{s.t.} & \quad f_1(x) = 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 - 127 \leq 0, \\
& \quad f_2(x) = 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 - 282 \leq 0, \\
& \quad f_3(x) = 23x_1 + x_2^2 + 6x_3^2 - 8x_7 - 196 \leq 0, \\
& \quad f_4(x) = 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0, \\
& \quad x^0 = (1, 2, 0, 4, 0, 1, 1)^T.
\end{align*}
\]

PROBLEM 6. Example s388 [34], the matrices a and b are defined in [34], omitted here.

\[
\begin{align*}
\min & \quad f_0(x) = -486x_1 - 640x_2 - 758xa - 776x_4 - 477xs - 707x_6 - 175x_7 - 619x_8 \\
& \quad - 627x_9 - 614x_{10} - 475x_{11} - 377x_{12} - 524x_{13} - 468x_{14} - 529x_{15}, \\
\text{s.t.} & \quad - b_i + \sum_{j=1}^{15} a_{ij}x_j^2 \leq 0, \quad i = 1, 2, \ldots, 10, \\
& \quad - b_i + \sum_{j=1}^{15} a_{ij}x_j \leq 0, \quad i = 12, 13, \ldots, 15, \\
& \quad - 0.5 \sum_{j=1}^{15} j (x_j - 2)^2 + 193.121 \leq 0, \\
& \quad x^0 = (0, 0, \ldots, 0)^T \in \mathbb{R}^{15}.
\end{align*}
\]

PROBLEM 7. Section 7.2, test problem 7 [35]

\[
\begin{align*}
\min & \quad f_0(x) = -x_1 + 0.4x_1^{0.67}t_3^{0.67}, \\
\text{s.t.} & \quad f_1(x) = 0.05882x_3x_4 + 0.1x_1 - 1 \leq 0, \\
& \quad f_2(x) = 4x_2x_4^{-1} + 2x_2^{0.71}x_4^{-1} + 0.05882x_2^{-1.3}x_3 - 1 \leq 0, \\
& \quad 0.1 \leq x_1, x_2, x_3, x_4 \leq 10, \\
& \quad x^0 = (7, 1, 0.5, 8)^T.
\end{align*}
\]

In all the tests, we set \(\varepsilon_- = 2\) (except for problem 6 with \(\varepsilon_- = 10\), we will note this later), \(\kappa = 2.5\), \(\beta = 0.9\), \(\alpha = 0.1\), and used the condition \(\|d_k^o\| \leq 10^{-6}\) as the stopping criterion. If \(I_k\) is empty, in order to avoid error, we let \(d(x^k) = d_o(x^k)\) in (2.11) and \(d_1(x^k) = 0\) in (2.11).

In formula (5.1), we set \(H_0 = E\), where \(E \in \mathbb{R}^{n \times n}\) is an identity matrix. In formula (6.1), we set \(\xi = 0.5\), \(\delta = 0.2\), and \(H_0 = E\). If \(L_k\) is empty, we let \(A_k A_k^T s^k\) be a zero vector with dimension \(n\).

For Algorithm JZX, the parameters were set as those in [18]. The numerical results of the proposed methods are given in Tables 1, 2, and 3. A detailed list about the approximate optimal solution \(x^*\) for Table 3 was reported in Table 4 (in fact, we
recorded all the details, but for Tables 1 and 2, we omitted here). The columns of these tables have the following meanings:

- \( N_i \): the number of iterations;
- \( N_{f0} \): the number of objective function evaluations;
- \( N_{g0} \): the number of objective gradient evaluations;
- \( N_f \): the number of constraint functions evaluations;
- \( N_g \): the number of constraint gradients evaluations.

### Table 1. Numerical results for Algorithm A by formula (5.1).

<table>
<thead>
<tr>
<th>Problem</th>
<th>Algorithm</th>
<th>( N_i )</th>
<th>( N_{f0} )</th>
<th>( N_{g0} )</th>
<th>( N_f )</th>
<th>( N_g )</th>
<th>( f_0(x^*) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Algorithm A</td>
<td>60</td>
<td>1797</td>
<td>60</td>
<td>14736</td>
<td>100</td>
<td>24.306214</td>
</tr>
<tr>
<td>(n = 8, m = 10)</td>
<td>Algorithm JZX</td>
<td>332</td>
<td>2014</td>
<td>333</td>
<td>18768</td>
<td>2664</td>
<td>24.326534</td>
</tr>
<tr>
<td>2</td>
<td>Algorithm A</td>
<td>8</td>
<td>66</td>
<td>8</td>
<td>216</td>
<td>12</td>
<td>(-44.113407)</td>
</tr>
<tr>
<td>(n = 4, m = 3)</td>
<td>Algorithm JZX</td>
<td>49</td>
<td>223</td>
<td>50</td>
<td>816</td>
<td>150</td>
<td>(-44.088089)</td>
</tr>
<tr>
<td>3</td>
<td>Algorithm A</td>
<td>12</td>
<td>81</td>
<td>12</td>
<td>356</td>
<td>25</td>
<td>(-2.000000)</td>
</tr>
<tr>
<td>(n = 3, m = 4)</td>
<td>Algorithm JZX</td>
<td>16</td>
<td>52</td>
<td>16</td>
<td>268</td>
<td>64</td>
<td>(-2.000000)</td>
</tr>
<tr>
<td>4</td>
<td>Algorithm A</td>
<td>6</td>
<td>19</td>
<td>6</td>
<td>115</td>
<td>11</td>
<td>(-2.000000)</td>
</tr>
<tr>
<td>(n = 2, m = 5)</td>
<td>Algorithm JZX</td>
<td>79</td>
<td>1140</td>
<td>74</td>
<td>9120</td>
<td>2240</td>
<td>(-2.000000)</td>
</tr>
<tr>
<td>5</td>
<td>Algorithm A</td>
<td>14</td>
<td>414</td>
<td>14</td>
<td>1700</td>
<td>17</td>
<td>(6.806301 \times 10^2)</td>
</tr>
<tr>
<td>(n = 7, m = 4)</td>
<td>Algorithm JZX</td>
<td>49</td>
<td>718</td>
<td>50</td>
<td>3068</td>
<td>200</td>
<td>(6.806756 \times 10^2)</td>
</tr>
<tr>
<td>6</td>
<td>Algorithm A</td>
<td>26</td>
<td>432</td>
<td>26</td>
<td>6840</td>
<td>147</td>
<td>(-5.821084 \times 10^2)</td>
</tr>
<tr>
<td>(n = 15, m = 15)</td>
<td>Algorithm JZX</td>
<td>560</td>
<td>9792</td>
<td>561</td>
<td>155280</td>
<td>8415</td>
<td>(-5.820569 \times 10^3)</td>
</tr>
</tbody>
</table>

### Table 2. Numerical results for Algorithm A by formula (5.2).

<table>
<thead>
<tr>
<th>Problem</th>
<th>Algorithm</th>
<th>( N_i )</th>
<th>( N_{f0} )</th>
<th>( N_{g0} )</th>
<th>( N_f )</th>
<th>( N_g )</th>
<th>( f_0(x^*) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Algorithm A</td>
<td>29</td>
<td>434</td>
<td>29</td>
<td>3648</td>
<td>74</td>
<td>(-44.113407)</td>
</tr>
<tr>
<td>(n = 8, m = 10)</td>
<td>Algorithm JZX</td>
<td>279</td>
<td>1140</td>
<td>280</td>
<td>9120</td>
<td>2240</td>
<td>(-2.000000)</td>
</tr>
<tr>
<td>2</td>
<td>Algorithm A</td>
<td>6</td>
<td>24</td>
<td>6</td>
<td>80</td>
<td>10</td>
<td>(-44.113407)</td>
</tr>
<tr>
<td>(n = 4, m = 3)</td>
<td>Algorithm JZX</td>
<td>104</td>
<td>407</td>
<td>105</td>
<td>1533</td>
<td>315</td>
<td>(-44.113407)</td>
</tr>
<tr>
<td>3</td>
<td>Algorithm A</td>
<td>14</td>
<td>96</td>
<td>14</td>
<td>405</td>
<td>26</td>
<td>(-2.000000)</td>
</tr>
<tr>
<td>(n = 3, m = 4)</td>
<td>Algorithm JZX</td>
<td>16</td>
<td>52</td>
<td>16</td>
<td>268</td>
<td>64</td>
<td>(-2.000000)</td>
</tr>
<tr>
<td>4</td>
<td>Algorithm A</td>
<td>20</td>
<td>575</td>
<td>20</td>
<td>2897</td>
<td>25</td>
<td>(-2.000000)</td>
</tr>
<tr>
<td>(n = 2, m = 5)</td>
<td>Algorithm JZX</td>
<td>8</td>
<td>59</td>
<td>8</td>
<td>330</td>
<td>40</td>
<td>(-2.000000)</td>
</tr>
<tr>
<td>5</td>
<td>Algorithm A</td>
<td>12</td>
<td>198</td>
<td>12</td>
<td>806</td>
<td>16</td>
<td>(6.806301 \times 10^2)</td>
</tr>
<tr>
<td>(n = 7, m = 4)</td>
<td>Algorithm JZX</td>
<td>351</td>
<td>3041</td>
<td>352</td>
<td>13520</td>
<td>1480</td>
<td>(6.808218 \times 10^2)</td>
</tr>
</tbody>
</table>

At the end of this section, we give a brief analysis for the numerical test results. First, by solving some typical practice problems from [33–35], we find the proposed Algorithm A is numerically effective (see Tables 1–3). Second, from Tables 1–3, we can see that the performances of Algorithm A are much better than Algorithm JZX for all problems only except Problem 4 in Table 2. Generally, the evaluations for iterations, objective function, objective gradient, constraint functions, and constraint gradients are much fewer than those by Algorithm JZX. On the other hand, the final objective values \( f(x^*) \) achieved by Algorithm A are more superior to Algorithm JZX for Problems 1, 2, 5, 6, 7, due to the fact that the search direction in Algorithm JZX is generated by a single generalized projection technique when the nonconvex QP in Algorithm JZX has no solution or its solution is dissatisfied.

As we mentioned above, we set \( \varepsilon_{-1} = 10 \) in Problem 6, and if we set \( \varepsilon_{-1} = 2 \), the evaluations of \( N_i, N_{f0}, N_{g0}, N_f, N_g \) will become 63, 2253, 63, 34650, 265 and 82, 5292, 82, 80460, 289 in Table 1.
Table 3. Numerical results for Algorithm A by formula (6.1).

<table>
<thead>
<tr>
<th>Problem</th>
<th>Algorithm</th>
<th>( N_i )</th>
<th>( N_{f_0} )</th>
<th>( N_g )</th>
<th>( N_f )</th>
<th>( N_g )</th>
<th>( f_0(z^*) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem 1</td>
<td>Algorithm A</td>
<td>32</td>
<td>912</td>
<td>32</td>
<td>7504</td>
<td>96</td>
<td>24.306209</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>332</td>
<td>2014</td>
<td>333</td>
<td>18768</td>
<td>2664</td>
<td>24.326534</td>
</tr>
<tr>
<td>Problem 2</td>
<td>Algorithm A</td>
<td>12</td>
<td>114</td>
<td>12</td>
<td>372</td>
<td>19</td>
<td>-44.13407</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>49</td>
<td>223</td>
<td>50</td>
<td>816</td>
<td>150</td>
<td>-44.08809</td>
</tr>
<tr>
<td>Problem 3</td>
<td>Algorithm A</td>
<td>9</td>
<td>33</td>
<td>9</td>
<td>152</td>
<td>18</td>
<td>-2.000000</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>16</td>
<td>52</td>
<td>16</td>
<td>268</td>
<td>64</td>
<td>-2.000000</td>
</tr>
<tr>
<td>Problem 4</td>
<td>Algorithm A</td>
<td>6</td>
<td>19</td>
<td>6</td>
<td>115</td>
<td>11</td>
<td>2.000000</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>8</td>
<td>59</td>
<td>8</td>
<td>330</td>
<td>40</td>
<td>2.000000</td>
</tr>
<tr>
<td>Problem 5</td>
<td>Algorithm A</td>
<td>19</td>
<td>459</td>
<td>19</td>
<td>1900</td>
<td>26</td>
<td>6.806301 \times 10^2</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>49</td>
<td>718</td>
<td>50</td>
<td>3068</td>
<td>200</td>
<td>6.806756 \times 10^2</td>
</tr>
<tr>
<td>Problem 6</td>
<td>Algorithm A</td>
<td>45</td>
<td>2074</td>
<td>45</td>
<td>31710</td>
<td>194</td>
<td>-5.821084 \times 10^3</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>560</td>
<td>9792</td>
<td>561</td>
<td>155280</td>
<td>8415</td>
<td>-5.820569 \times 10^3</td>
</tr>
<tr>
<td>Problem 7</td>
<td>Algorithm A</td>
<td>69</td>
<td>2419</td>
<td>69</td>
<td>24840</td>
<td>102</td>
<td>-5.739820</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>176</td>
<td>528</td>
<td>177</td>
<td>7040</td>
<td>1700</td>
<td>-5.438387</td>
</tr>
</tbody>
</table>

Table 4. The approximate optimal solution \( z^* \) for Table 3.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Algorithm</th>
<th>( z^* )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem 1</td>
<td>Algorithm A</td>
<td>(2.171996, 2.363683, 8.773926, 5.095985, 0.990655, 1.430574, 1.321644, 9.828726, 8.280991, 8.375927)^T</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>(2.173586, 2.359300, 8.773038, 5.095608, 0.983695, 0.417117, 1.325201, 9.831115, 8.280973, 8.372555)^T</td>
</tr>
<tr>
<td>Problem 2</td>
<td>Algorithm A</td>
<td>(-0.019533,0.855079,2.019151,-1.085252)^T</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>(-0.017094,0.858481,2.015147,-1.085871)^T</td>
</tr>
<tr>
<td>Problem 3</td>
<td>Algorithm A</td>
<td>(0.000000,0.000000,2.000000)^T</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>(0.000000,0.000000,2.000000)^T</td>
</tr>
<tr>
<td>Problem 4</td>
<td>Algorithm A</td>
<td>(1.000000,1.000000,2.000000)^T</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>(1.000000,1.000000,2.000000)^T</td>
</tr>
<tr>
<td>Problem 5</td>
<td>Algorithm A</td>
<td>(2.330499,1.951372,-0.477541,4.365726, -0.624487,1.038131,1.594227)^T</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>(2.332780,1.951485,-0.470221,4.363831, -0.624439,1.029458,1.595017)^T</td>
</tr>
<tr>
<td>Problem 6</td>
<td>Algorithm A</td>
<td>(0.628838,1.433100,1.462596,0.731333,0.786143, 1.204860,-1.143399,1.061111,-0.133893,1.182010, 0.969177,-0.845019,0.481225,-0.339861,0.685890)^T</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>(0.627401,1.431916,1.462616,0.730864,0.786646, 1.204936,-1.141505,1.062668,-0.133855,1.180699, 0.970820,-0.851013,0.483548,-0.339533,0.687983)^T</td>
</tr>
<tr>
<td>Problem 7</td>
<td>Algorithm A</td>
<td>(8.130072,0.615366,0.564044,5.636208)^T</td>
</tr>
<tr>
<td></td>
<td>Algorithm JZX</td>
<td>(7.717226,0.617161,0.574936,8.875507)^T</td>
</tr>
</tbody>
</table>

and Table 3 respectively. But we should point out here, for other problems, the alternative of this parameter was slightly or no different to the numerical results (we have tested).

Problem 6 by formula (5.1) and (6.1) implemented well, but failed to (5.2), Problem 7 also failed to formula (5.1) and (5.2). Following the referees advice, we tested some practical problems in [35], Table 3 shows that Problem 7 performed a satisfied iteration for formula (6.1), but it is a pity, some problems from [35] failed to the three formulas (5.1), (5.2), and (6.1), such as test problem 7.2.4 and 7.2.9.
7. CONCLUDING REMARKS

We have presented here a new SQP feasible descent algorithm for solving nonlinear optimization problems with nonlinear inequality constraints. The algorithm starts with a feasible point, generates a master direction by solving a quadratic program (which always has feasible solution). With some modification on the master direction by two explicit formulas, the algorithm generates a feasible descent direction and a height-order correction direction (used to avoid the Maratos effect), then, performs a curve search to obtain the next iteration point. Due to the introduction of the new height-order correction technique (2.11), (2.12), under mild conditions without the strict complementarity, we proved that the new algorithm possesses global, superlinear, and even quadratical convergence properties. Finally, an efficient implementation of the proposed algorithm was reported. We conjecture that the technique introduced in this paper can be used to modify some other SQP algorithms (such as [12,14,15,18-20]), such that the strict complementarity may be avoided.

REFERENCES
2. S.P. Han, Superlinearly convergent variable metric algorithms for general nonlinear programming problems, Mathematical Programming 11, 263–282, (1976).


31. J.B. Jian, Researches on superlinearly and quadratically convergent algorithms for nonlinearly constrained optimization, Ph.D. Thesis, School of Xi’an Jiaotong University, Xi’an, China, (2000).


